SALIENCY IN AUDIO AND VISUAL SIGNALS

BY

KAI-HSIANG LIN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Doctoral Committee:

Professor Thomas S. Huang, Chair
Professor Mark Allan Hasegawa-Johnson
Associate Professor Feng Liang
Professor Zhi-Pei Liang

ABSTRACT

This dissertation studies saliency and its applications in audio and visual signals. For each portion of the signal, its saliency means the likelihood of attracting bottom-up attention in the perception process. In computer vision, image saliency is described by local contrasts of features. Each local patch is divided into center and surround regions based on their spatial distance to the center of the patch. Differences of features between the two regions are used as the saliency estimation. The existing saliency framework used different methods to measure the center and surround difference and was able to detect image saliency reasonably well. Although these frameworks are suitable for detecting image saliency, they may not be suitable for detecting saliency in signals containing temporal information such as image sequences. In this dissertation, we propose a new saliency framework based on outlierness of the center region comparing to the surround region. Specifically, the surround region is divided into several subregions and the feature distances between the center and surround subregions are computed. The k^{th} nearest distance is used as the outlierness to estimate the saliency of the center region. Based on this framework, we propose a novel image saliency detection algorithm and compare its performance with existing algorithms. This framework is also successfully applied to image sequences to detect foreground in dynamic scenes. Besides foreground detection, we also propose two new applications of saliency detection on images and audio. First, we propose an algorithm of a license plate detection inspired by the observation of license plate being salient. Characters are first located using a segmentation of the intensity saliency map. Some saliency-based features are extracted on the neighborhood of the characters to detect license plates accurately. Second, we propose an algorithm maximizing the saliency of audio spectrograms. This audio visualization enables efficient audio-visual browsing for faster-than-real-time human acoustic event detection. Visual saliency

has been used as a metric to evaluate different information visualizations in the literature. In our work we not only formulate a new saliency-based metric for information visualization but also use this metric to automatically enhance the spectrogram.

To my family, for their love and support

ACKNOWLEDGMENTS

I would like to express my sincerest gratitude to my adviser, Prof. Thomas S. Huang, for his invaluable guidance, support, encouragement and care throughout the course of my graduate study. I am thankful to every member of my committee for their valuable comments and suggestions in my dissertation work. I am particularly grateful to Prof. Mark Hasegawa-Johnson for co-advising much of the work I have done.

I thank all my collaborators: Dirk Bernhardt-Walther, Xiaodan Zhuang, Ming Yang, Kyungtae Kim, Xu Zhao, Usman Tariq, Zhaowen Wang, Jiangping Wang, Pooya Khorrami and Xiaoyuan Yu. I feel honored and blessed to have had the opportunity to work with and learn from them. I appreciate all the members of the Image Formation and Processing (IFP) group at Beckman Institute and many other friends at the University of Illinois at Urbana-Champaign. Their help and support made my PhD life exciting and enjoyable.

Last but not least, I give special thanks to my family: my parents, my parents-in-law, my wife Ting-Yu and my daughter Jane, for their unconditional spiritual support, patience, encouragement and company throughout several challenging times during my studies. It is impossible to imagine life without their support.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Humans have the remarkable ability of interpreting complex scenes instantly; on the contrary, computer vision has so far been unable to achieve similar results. Partly, this is due to limitations in processing power. Humans cope with similar limitations by quickly scanning the scene and then only paying attention to salient (conspicuous) parts. Here, saliency means the ability of attracting bottom-up (i.e., stimulus-driven as opposed to pre-meditated) attention in the visual perception process. The saliency detection algorithm has been used in many applications such as video compression [1], image segmentation [2], and advertising design [3].

Saliency algorithms can be reduced in almost all cases to local contrasts in some sorts of features. In computer vision, the local contrasts can be computed using center-surround differences. Overall saliency is then coded in a spatially organized saliency map. There have been many saliency detection algorithms proposed and each of them tried to model saliency from different points of view. For example, cognitive models are inspired by psychological or neurophysiological findings and information theoretic models assume salient regions have more information then non-salient regions in an image [4].

Although many existing saliency models detect image saliency reasonably well, these models may not be suitable to detect saliency in signals containing temporal information. In this dissertation, we propose a saliency framework which is suitable for detecting saliency not only in images but also signals containing temporal information such as image sequences. In this dissertation we model saliency as an outlier in a neighborhood of the signal and explore new applications of saliency on audio and visual signals.

The rest of the dissertation is organized as follows. In Chapter 2, we propose a novel framework of saliency detection based on local outlierness using the k^{th} nearest neighbor distance. This framework could be used to detect saliency in both spatial and temporal neighborhoods. Three algorithms are

proposed to apply saliency to images (Chapter 3), image sequences (Chapter 4), and audio (Chapter 5). In Chapter 3, we propose an algorithm to detect license plates using new features of saliency. In Chapter 4, we extend spatial saliency to temporal saliency to detect foreground in image sequences with dynamic scenes. In Chapter 5, we propose a saliency-maximized audio visualization for efficient audio-visual browsing.

CHAPTER 2

SALIENCY BASED ON OUTLIER DETECTION

In this chapter, we propose a novel saliency algorithm based on outlier detection. We formulate the saliency of each image patch as the outlier tendency of its center region by comparing it to the surround region. For an input image, we extract features on image patches and compute the patches' outlierness based on the feature distance. In each image patch, the surround region is divided into N subregions and feature distances between these subregions and the center region are computed. Saliency maps of individual features are generated using the k^{th} nearest distance of each image patch. Saliency maps of different features are combined using normalization and summation to generate the final saliency map. The proposed algorithm is compared with two popular saliency detection algorithms and achieves better performance.

2.1 Introduction

In neuroscience, saliency means the ability to attract bottom-up attention in the visual perception process. In Figure 2.1 (a), a bright bar is the most salient among the dark bars, for example. A common model of bottom-up saliency is inspired by the center-surround difference mechanisms in the early stages of biological vision. From the evidence of electrophysiology, typical visual neurons in visual receptive fields are arranged into a central disk, the center, and a concentric ring, the surround, each region responding oppositely to stimuli [5]. The center-surround receptive field organization is illustrated in Figure 2.1 (b) (different colors represent opposite responses).

In computer vision, there have been many saliency detection algorithms proposed [6]. There are two popular algorithms which are highly related to the proposed method. One algorithm was proposed by Itti et al. and the other one was proposed by Bruce and Tsotsos.

Itti et al. modeled the center-surround difference by a difference of Gaussian (DoG) and proposed an algorithm to compute a saliency map proportional to local feature contrasts [7]. A DoG consists of two 2D Gaussian functions with different variances:

$$\mathrm{DoG}[n_1, n_2] = \frac{1}{2\pi}\left(\frac{e^{-(n_1^2+n_2^2)/2\sigma_c^2}}{\sigma_c^2} - \frac{e^{-(n_1^2+n_2^2)/2\sigma_s^2}}{\sigma_s^2}\right) \tag{2.1}$$

This DoG function is parameterized by two σ's. The first Gaussian has the smaller σ_c; the second has the larger σ_s. (Subtracting these Gaussians approximates a Mexican hat function.) Filtering an image with the central Gaussian, the first term in Eq. 2.1, averages the features within approximately σ_c pixels of the output pixel. The surround Gaussian, the second term, computes a similar average for σ_s. Thus center-surround difference estimates how much the pixels near (n_1, n_2) stand out, relative to those in the surrounding disc of radius σ_s.

(a) An example of image saliency. (b) Center-surround visual receptive field.

Figure 2.1: Image saliency.

Bruce and Tsotsos modeled the saliency as the self-information of an image patch comparing to its surround region [8]. For an image patch Y, saliency measure based on self-information is given by $-log(p(Y))$. p is a probability function describing the distribution of some feature set over the surround region of Y. $p(Y)$ describes the likelihood of observing Y based on the feature distribution of its surround region. If $p(Y)$ is small, Y has both high self-information and a high saliency score.

These two saliency detection algorithms have some limitation respectively. In the first algorithm, using DoG to compute saliency implicitly utilizes the weighted mean of the feature to describe the center and surround regions. This description is more suitable for the center region which has a smaller

area. For the surround region which usually has a much larger area, the weighted mean of the feature might not be able to characterize the region effectively. In the second algorithm, measuring self-information requires a reasonably good estimation of the feature distribution. Estimating the feature distribution could be very challenging when the sample size in the surround region is not large enough.

In this chapter, we propose to model the saliency as the outlierness of the center region by comparing it to the surround region. The outlierness is computed using the feature distance between a center region and its k^{th} nearest neighbor (KNN) in the surround region. In comparison to other existing outlier detection methods [9], the KNN approach works relatively well when the sample size is small.

2.2 Saliency based on outlier detection

There are three steps in our algorithm: (1) feature extraction; (2) outlier analysis; and (3) feature combination. A visual representation of our algorithm is shown in Figure 2.2.

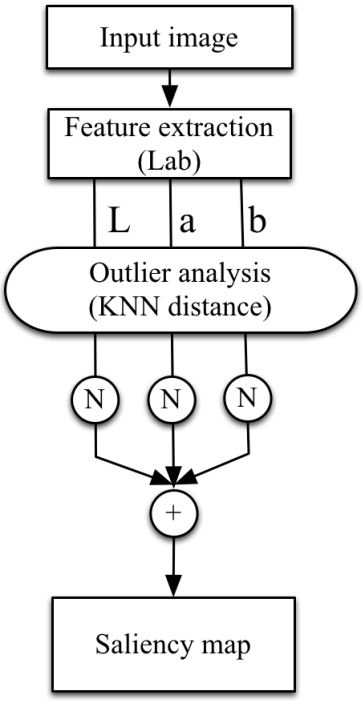


Figure 2.2: The framework of the proposed saliency detection algorithm.

2.2.1 Feature extraction

We choose Lab color space as the intensity and color features because of its perceptual uniformity [10]. Perceptually uniform means that a change of the same amount in a color component value should produce about the same change in visual perception. For an input image, each pixel is described by one lightness component (L), and two chromatic opponent components: red/green (a) and blue/yellow (b).

After converting an input image to three feature channels, patch-based features are computed on all three feature maps:

$$\begin{aligned} h_{g,c}(x,y) &= \sum_m \sum_n G_c(m,n) W_{g,c}(x+m, y+n), \qquad g \in \{L,a,b\} \\ h_{g,s_i}(x,y) &= \sum_m \sum_n G_s(m,n) W_{g,c}(x_{s_i}+m, y_{s_i}+n), \qquad i = 1,...,N \end{aligned} \tag{2.2}$$

where h(x,y) is the weighted average of the subpatch $W(x,y)$ with the center at location (x,y). G_c and G_s are Gaussian weighting functions with $\sigma_c = 4$ and $\sigma_s = 21$ respectivly. (x_{s_i}, x_{s_i}) is the center of the surround subregion s_i. The distances between the center of the center region and the centers of the surround subregions are 41. Figure 2.3 illustrates the geometric pattern of the center region c and surround subregions s_i's. Note also that Eq. (2.2) could be computed efficiently by separable Gaussian convolution.

2.2.2 Outlier analysis

We propose to compute the saliency of each pixel based on the outlierness of the image patch on this pixel. The outlierness is measured using the distance of the k^{th} nearest neighbor. We formulate the saliency S as:

$$\begin{aligned} d_g(x,y,i) &= |h_{g,c}(x,y) - h_{g,s_i}(x,y)| \\ D_{g,x,y} &= \{d_g(x,y,i) | i = 1,...,N\} \\ d_{g,sort}(x,y,j) &= sort\,\{D_{g,x,y,i}\} \\ S_g(x,y) &= d_{g,sort}(x,y,k) \end{aligned} \tag{2.3}$$

where $d_g(x,y,i)$ is the feature distance between the center region c and surround subregions s_i's at location (x,y). The g is the same feature index as

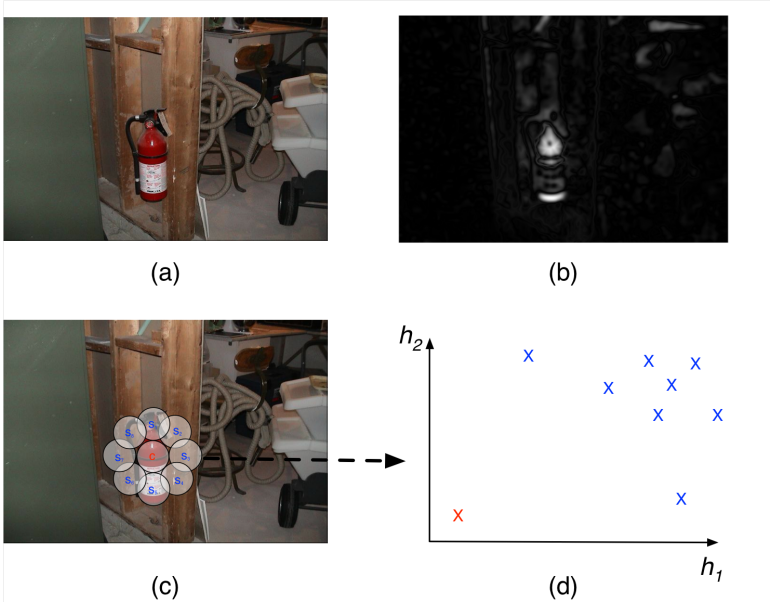


Figure 2.3: Illustration of saliency detection based on KNN distance: (a) input image; (b)S_a: saliency map of red/green component; (c) geometric pattern of the center region and surround subregions; (d) feature samples of the center and surround subregions in the feature space.

in Eq. (2.2). $D_{g,x,y}$ is the set of distances. $N = 8$ is the number of surround subregions. $d_{g,sort}(x, y, j)$ is the sorted distance in ascending order. $S_g(x, y)$ is the magnitude of the saliency. The distance of the second nearest neighbor ($k = 2$) is used in the experiments. After outlierness analysis, we have three saliency maps of different features.

2.2.3 Feature combination

We combine the saliency maps of different features into a single saliency map, normalizing before every summation with a nonlinear operator $N(\cdot)$. This operator restores a similar dynamic range to the saliency maps of different visual modalities, and also enhances strong local peak response. It is implemented with a large 2D DoG filter. This is followed by positive rectification [11].

The combined saliency map $S(x, y)$ is the mean of the normalized maps for the lightness and two chromatic opponent components:

$$S(x, y) = (\sum\nolimits_g N\left(S_g(x, y)\right))/3 \tag{2.4}$$

In most of the algorithms of image saliency, some kind of smoothing has been included. In the experiments, we post-process S by the Gaussian smoothing with $\sigma = 25$. Similar to [12], all the pixels connected to the image borders are set to zero based on the Gestalt principle of figure-ground segregation.

2.3 Experiments

We evaluate the proposed algorithm by measuring how well it can predict human eye fixations [13]. We compare the proposed algorithm with two popular saliency algorithms. The saliency maps produced by different algorithms are treated as detectors of fixation points. The intensity of the saliency map is the prediction score of fixated pixel locations. In this setup, a receiver operating characteristic (ROC) curve is generated for each algorithm and the area under ROC curve (AUC) is calculated.

In the experiments we use the TORONTO dataset which is widely used for saliency model comparison [8]. It contains 120 natural color images from indoor and outdoor environments. Images were presented to 20 subjects in random order for four seconds. There was a gray mask for 2 seconds between each pair of images.

Table 2.1 shows the accuracy of saliency detection on the dataset in comparison with two popular saliency algorithms [7, 8]. It can be seen that the proposed method outperforms the other two methods in this dataset. Examples of qualitative comparison between fixation density maps of human subjects and saliency maps of our method is illustrated in Figure 2.4.

Table 2.1: Performance comparison of saliency detection on the TORONTO dataset.

Saliency model	Itti [14]	AIM [8]	This work
ROC area	0.7277	0.7565	0.7856

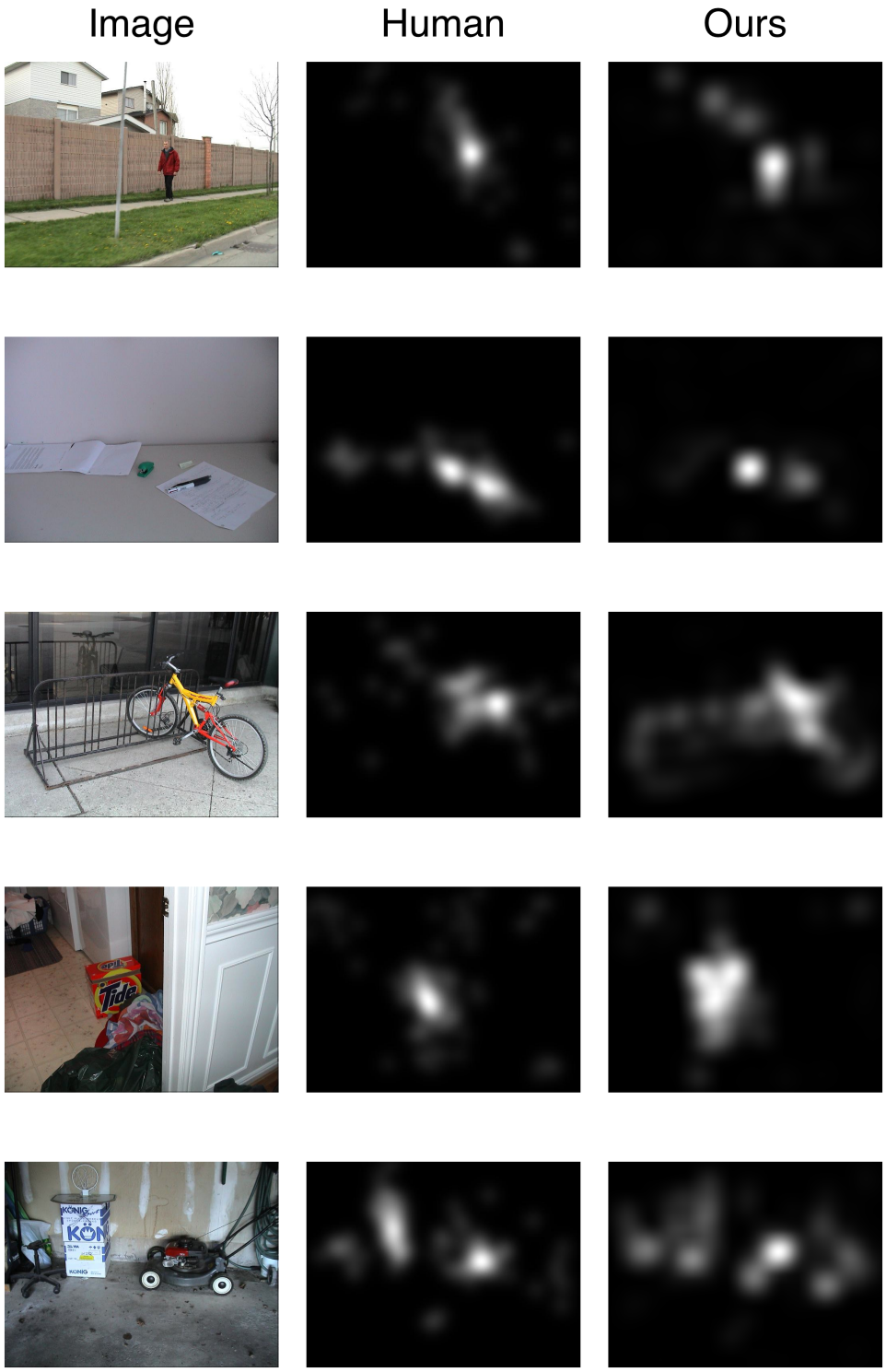


Figure 2.4: Visual comparison of human model and the proposed saliency model over samples from the TORONTO dataset.

CHAPTER 3

SALIENCY IN IMAGES: LICENSE PLATE DETECTION

Inspired from the observation that license plates are very salient to human visual perception, we propose a novel license plate detection algorithm based on image saliency in this chapter. The proposed algorithm consists of two parts. The first part segments out the characters on a license plate using an intensity saliency map with a high recall rate. The second part applies a sliding window on these characters to compute some saliency-related features to detect license plates. We test the robustness of our algorithm by applying it on a mixed dataset with high diversity collected from four datasets. The mixed dataset has 1024 images composed of license plates of all states of the United States. We achieve a detection rate of 90% with False Positive Per Image (FPPI) = 12%. The detection box given by our algorithm has high precision, which will be very helpful for many applications such as license plate recognition.

3.1 Introduction

In the past the applications of license plate detection were mostly related to intelligent transportation systems, for example, security control and traffic monitoring. Recently this topic has gained more popularity in privacy protection as the number of images and videos on the Internet is exponentially increasing. One of the most famous applications of privacy protection is to blur license plates in the Google Street View.

There have been a lot of license plate detection algorithms proposed in the past. In [15], the Hough transform was used to detect boundaries of license plates. In [16], a series of gray-level morphological operations were applied to the input image to detect vertical edges of license plates. After detecting features of license plates, these two algorithms applied some predefined rules

to extract the regions of license plates. Besides the rule-based algorithms, there are some learning-based algorithms by treating license plate detection as a binary classification problem. In [17], a Support Vector Machine (SVM) classifier was applied to color texture to detect license plates. A license plate detection algorithm modified from a face detection algorithm using AdaBoost on Harr-like features was proposed in [18]. Recently, the covariance descriptor was also employed with a neural network to detect license plates [19].

Although license plate detection is a well-studied problem, detecting license plates with different styles, scales, poses, illumination, or partial occlusion is still a challenging task. Because we observe that the license plates are designed to be very salient to visual perception, we develop a license plate detector based on image salinecy. Here, saliency means the ability of attracting bottom-up (i.e., stimulus-driven as opposed to pre-meditated) attention in the visual perception process. In our algorithm the intensity saliency map is used to detect the characters of a license plate precisely with a high recall rate, and the following license plate detection with saliency-related features effectively eliminates the false positives. Our experimental results on a mixed dataset with high diversity show that our algorithm is robust.

3.2 License plate detection algorithm

There are five steps in our algorithm: (1) compute the saliency map using center-surround difference; (2) detect the characters of the license plates; (3) apply a sliding window on each character and compute features of each window; (4) estimate the likelihood of each window as a license plate and make final decision; and (5) pick the window with the largest likelihood value as the detection box for each cluster of overlapped bounding boxes. The procedure of our algorithm is shown in Figure 3.1.

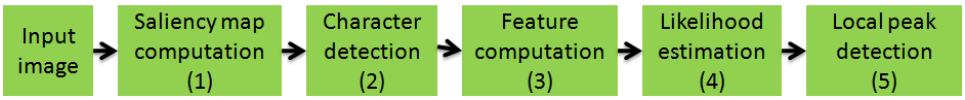


Figure 3.1: The framework of our license plate detector.

3.2.1 Saliency map computation

Saliency is usually modeled as the center-surround difference of different kinds of features such as intensity, orientation, and color [7]. Here we only use intensity saliency and thus all the input images will be converted into grayscale images. The saliency map of intensity can be derived by convolving the image with a Difference of Gaussian (DoG). A DoG consists of two 2D Gaussian functions with different variances:

$$\mathrm{DoG}(x,y) = \frac{1}{2\pi}\left(\frac{1}{\sigma_1^2}e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{\sigma_2^2}e^{-\frac{x^2+y^2}{2\sigma_2^2}}\right) \tag{3.1}$$

where the Gaussian functions with σ_1 corresponds to the central region, and the other Gaussian function corresponds to the surround region ($\sigma_1 < \sigma_2$). Gaussian pyramids of input images are built to efficiently compute the DoG of different scales. The Gaussian kernel used in this algorithm is [1 5 10 10 5 1]/32. The value of σ_1 should be comparable to the width of the line of characters which is usually very thin. Larger σ_1 can filter out some noise but may blur the characters, which will cause problems when using connected components to segment out the characters in license plates of a very small scale. Because these two Gaussian functions are used to estimate the average properties of the central and surrounding regions, the ratio of the variances of these two Gaussian functions is usually large. We used the first layer and fouth layer of the Gaussian pyramid as our center and surround layers in this algorithm.

There are two kinds of saliency, on-off and off-on saliency, caused by the brighter center within dark surround or the other way around. For each image after the DoG operation, the on-off saliency map is the positive region and the off-on saliency map is the negative region. All characters of a single license plate should be either dark or bright. We process these two kinds of saliency maps separately to detect dark characters with bright surround and bright characters with dark surround in license plates.

3.2.2 Character detection

The characters in a license plate can provide the precise information of the scale and location of the potential license plate. We use them as our basic

elements to detect license plates. On the saliency map we use connected components to segment out the possible character regions and then measure the geometric properties of the segments as features. (From our observation, license plates usually have some salient geometric properties.) An object will be detected as a character if its aspect ratio and scale are in a specific range. The reason we choose the aspect ratio is that it is scale invariant.

Most of the time the characters are separated from other segments in the saliency map, but sometimes the top and bottom of the characters might be connected to other objects such as the frame of the license plate. To overcome this problem, we horizontally cut the saliency map sequentially. A connected component operation is applied after each cut, and we only keep the segment having the aspect ratio within a specific range. For the datasets we use, the range is between 1.6 and 4. This range is loose and we may get many false alarms at this stage. The character detection step is designed to have a high recall rate, and false positives will be eliminated in the following stages. Figure 3.2 is an example of character detection in our algorithm.

(a) Input image (b) Character detection

Figure 3.2: Input image and character detection on its on-off saliency map.

Figure 3.2(a) is the input image and Figure 3.2(b) is the result of character detection from the on-off saliency map. We can observe that there are many false positives in Figure 3.2(b), especially around the grill in front of the car.

3.2.3 Feature computation

In order to eliminate the high false positives of character detection, we compute some features of the windows around the characters and use them to detect license plates with a low false positive rate.

A license plate is a flat rectangular region which contains several characters with similar saliency and of almost the same height. Based on this consistency, we apply a sliding window on each character with the same height as the corresponding character and compute a set of features for each window. When computing features of each window, we only consider characters overlapped by the window and all these characters must have similar heights and saliency. We initially set the window width equal to four times the height and then modify the width so that the window exactly contains all possible characters. This width/height ratio comes from the maximum of the aspect ratios of license plates in our dataset.

Based on the general properties of license plate, we design several features of license plates which can be divided into two categories. One is inspired from saliency including saliency of characters, saliency of the windows, and entropy. The other is designed based on our observation of license plates and not related to saliency including character quantity, overlap, aspect ratio, and location. The definition of each feature is explained as follows:

1. The saliency of characters is the average intensity of the involved characters on the saliency map. Generally speaking, the saliency feature of a window containing a license plate is usually relatively large.

2. The saliency of windows is the center-surround area ratio of a region of potential license plates. A license plate must be salient in the context. In other words, there should not be many windows similar to license plates spreading in a region. This rule will eliminate noisy regions such as trees. We first connect all windows of the same scale if they are close to one other, and the saliency of a window is the area ratio of this window and the region it is connected to.

3. Entropy is also a popular feature to detect image saliency [20]. The entropy feature used this algorithm is the entropy of the (normalized) histogram of oriented gradients (HOG). We first compute the HOG of a window and compute the entropy of this distribution. Because we quantize edges into eight directions, the entropy of each window will be between zero and three bits. As there should be a certain amount of information in the license plate, the entropy of a window containing a license plate should not be too small. This feature is useful to eliminate

windows on parallel lines such as the grill in front of the car in Figure 3.2(b).

4. The character quantity is the number of characters contained in a window. This number must be within some range.

5. If a window contains a license plate exactly, the bounding boxes of the characters should overlap most regions of the window. The overlap feature we use is similar to the criterion used in segmentation accuracy evaluation. Let the region W be the sliding window and region C be the bounding boxes of candidate characters at least partially overlapping with the window. We define the overlap feature of each window as $P(C;W) = \frac{|C \cap W|}{|W|} + \frac{|C \cap W|}{|C|}$, where $|\cdot|$ is the operation of computing the segment area.

6. The aspect ratio is the ratio of the width and height of a window. This feature can be used to prevent the algorithm from choosing the license plates with high overlap but is too narrow or too wide.

7. We use the coordinates of the center of the window divided by the width and height of the image as our feature of location. License plates are not equally likely to located at every location of the image. In general, most license plates are located at the lower part of the image. This feature helps to decrease false positives similar to license plates such as street signs.

3.2.4 Likelihood estimation

In order to combine all the features described above, we compute the likelihood of each window through estimating the joint conditional probability of these features given that a window contains a license plate. The window with a large likelihood value will be detected as a license plate.

Assuming the features comply with the naive Bayes assumption, we have

$$P(f_1, ..., f_7|\text{license plate}) = \prod_{i=1}^{7} P(f_i|\text{license plate}) \tag{3.2}$$

where f_i is the ith feature. In our experiments, the likelihood of "character

quantity" is uniformly distributed in the interval $[3, 9]$, and "saliency of the window" is uniformly distributed in the interval $[1, 10]$. The likelihood of all other features are estimated by the histogram of the ground truth data in the dataset.

3.2.5 Local peak detection

For each character there are many bounding boxes around it. Thus there should be a cluster of bounding boxes around the license plate. In order to pick the best detection box, we do connected components analysis again and within each blob only the bounding box with the highest likelihood will be retained.

3.3 Experiments

3.3.1 Evaluation dataset

We test our algorithm on a mixed dataset with license plates of high diversity collected from the four datasets: (i) UCSD/Calit2 [18]; (ii) Caltech cars;[1] (iii) UIUC/IFP; and (iv) Flickr.[2] There are 1024 images and 1028 license plates in the mixed dataset, including license plates of all states of the United States. Some images have no license plates and some have multiple license plates. There are more than five license plates of any state, and most states have more than two styles of license plates. The dataset could be divided into two categories: (1) Constrained data: this category includes the datasets of (i) and (ii). Almost all license plates in this category are of similar scales, resolutions and locations. These license plates contain six to seven clear characters and most of them are license plates of California. (2) Unconstrained data: this category includes the datasets of (iii) and (iv). The license plates in this category have different resolutions, scales, backgrounds, illuminations and angles. The dataset (iv) is the most challenging because the license plates in this data base have the largest diversity. Furthermore, many of the license plates downloaded from dataset (iv) are special, because

[1] http://www.vision.caltech.edu/html-files/archive.html
[2] http://www.flickr.com/

people tend to collect special license plates and upload them to their online album. For example, unlike almost all license plates in the other dataset containing at least six characters, some license plates downloaded from dataset (iv) contain fewer than three characters. The numbers of images and license plates of each dataset are listed in Table 3.1.

Table 3.1: The number of images and license plates in each dataset.

Dataset	(i)	(ii)	(iii)	(iv)	Total
# of Images	126	291	332	275	1024
# of LPs	124	287	342	275	1028

3.3.2 Results

We apply our algorithm to the mixed dataset and use fivefold cross-validation to evaluate the performance of our algorithm. Because the most important information of a license plate is its text and in most applications character recognition will be applied next, we define our ground truth as the bounding boxes of the text in the license plates. The correctness of detection is the same criterion used in PASCAL (Intersection/Union ≥ 0.5). Each predicted bounding box and ground truth bounding box can be matched at most once. To measure the detection performance, we use detection rate vs. False Positive Per Image (FPPI) because it is natural for human interpretation [21]. The proposed algorithm achieves a detection rate of 90% with FPPI = 12%. The general performance of the algorithm is shown by the red curve in Figure 3.3. We also study the effect of the features related to saliency by comparing the performance with and without non-saliency-related features. As shown in Figure 3.3, the performance with saliency-related features is significantly better than the performance without saliency-related features, especially when the FPPI is less than 0.1. The relative miss rate reduction at some specific FPPI values is listed in Table 3.2.

Table 3.2: Relative miss rate reduction at some specific FPPI values.

FPPI	0.05	0.1	0.2
Relative miss rate reduc.	26%	28%	12%

We illustrate five examples of the detection results in Figure 3.4. These

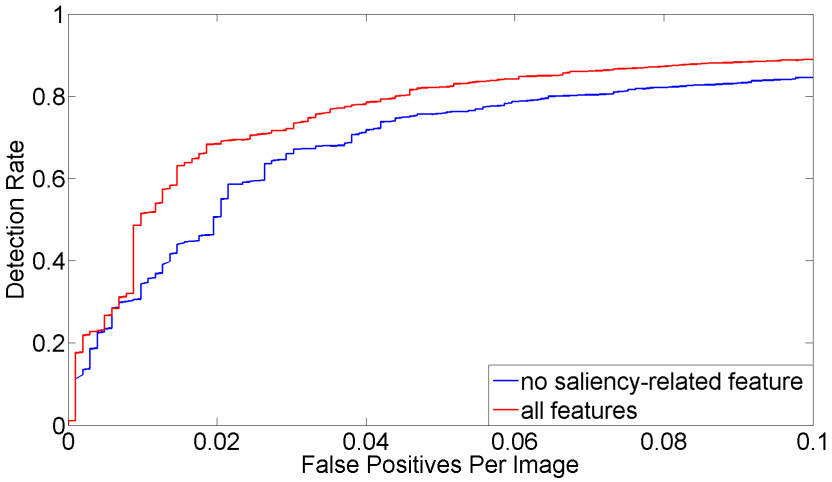


Figure 3.3: Performance comparison of detection with/without saliency-related features. The red curve represents the performance of the proposed algorithm using all the features mentioned in Section 3.2.3. The blue curve represents the performance without using saliency-related features.

examples demonstrate the diversity of our test dataset. Figure 3.4 (a) shows an enlarged license plate at the right-bottom corner. This is used to illustrate that the proposed algorithm can obtain a high-precision bounding box. Being able to detect a high-precision bounding box is important. For example, we need to recognize the characters after detecting a license plate in many applications. Optical character recognition (OCR) on natural scene images is a non-trivial task, and a precisely detected bounding box can help this process a lot. The main reason of this high precision that our algorithm achieves comes from the good character segmentation results of the first stage. An correct character segmentation can provide accurate information of the location and scale of license plate.

The accuracy of the proposed method is compared with a state-of-the-art method on the public Caltech cars dataset [22]. This method detects license plates based on SIFT features. The comparison the two methods is shown in Table 3.3. It can be seen that the proposed method significantly outperforms the other method in this dataset.

(a)

(b)

(c)

(d)

(e)

Figure 3.4: Sample detection results. The detection box is marked as red rectangular. (a) An example demonstrate the precision of the detection box. The detection box is enlarged and placed at right-bottom corner (DB (i)). (b) License plate is blurred and partially occluded (DB (ii)). (c) License plate under low illumination (DB (iii)). (d) Tilted license plate (DB (iii)). (e) Image taken through the windshield (DB (iv)).

Table 3.3: Performance comparison of license plate detection on the Caltech cars dataset.

	PVW [22]	This work
Precision	0.955	**0.991**
Recall	0.848	**0.938**
F score	0.898	**0.963**

CHAPTER 4

SALIENCY IN IMAGE SEQUENCES: FOREGROUND DETECTION

In this chapter, we propose a novel saliency-based algorithm to detect foreground regions in highly dynamic scenes. We first convert input video frames to multiple patch-based feature maps. Then, we apply temporal saliency analysis to the pixels of each feature map. For each temporal set of co-located pixels, the feature distance of a point from its k^{th} nearest neighbor is used to compute the temporal saliency. By computing and combining temporal saliency maps of different features, we obtain foreground likelihood maps. A simple segmentation method based on adaptive thresholding is applied to detect the foreground objects. We test our algorithm on images sequences of dynamic scenes, including public datasets and a new challenging wildlife dataset we constructed. The experimental results demonstrate the proposed algorithm achieves state-of-the-art results.

4.1 Introduction

Detecting moving objects in an image sequence is an important step in intelligent video analysis. There have been many algorithms developed for foreground object detection [23, 24]. However, detecting foreground objects from cluttered and highly dynamic background is still a challenging task. In a realistic outdoor image sequence, there are many potential factors to make foreground detection difficult. These factors include changing illumination, camouflage, moving background objects, and shadowing.

For videos with highly dynamic scenes, background subtraction based on different saliency models were proposed. In [25], discriminant saliency was computed on the dynamic texture model of spatio-temporal patches to generate the saliency map. Under this model, a location was assigned to the background if its saliency score was below a certain threshold. In [26],

temporal saliency of bag-of-words features as well as spatial saliency of color histograms and Local Binary Patterns (LBP) were used in a graph-cut framework to do foreground object segmentation.

In this chapter we propose a novel background subtraction algorithm using temporal saliency. The saliency of a pixel is an estimate of how much it stands out relative to its neighbors. In most image saliency algorithms, saliency is modeled by center-surround difference [6]. The center and surround regions are defined based on spatial proximity. The center region is a small neighborhood of the location of interest while the surround region is a larger neighborhood surrounding the center region. This definition is reasonable for detecting saliency in a single image, but may not be suitable for image sequences containing a temporal component.

The presence of dynamic content in image sequences suggests that a more robust representation is needed when computing saliency. Essentially, if the center region has a feature distribution that is drastically different from the feature distribution in the surround region, the center region is deemed salient. Unfortunately, with highly dynamic scenes, the feature distribution of co-located pixels in fixed temporal windows becomes more difficult to model and causes the parametric model used during saliency estimation to degrade.

Inspired by the work of outlier detection using the distance of the k^{th} nearest neighbor [27], we address the aforementioned problem by making a given pixel be the central region and its k nearest neighbors in a feature space be the surround region. Then, for each temporal set of co-located pixels in a sequence, we calculate the feature distance of a point from its k^{th} nearest neighbor to estimate the temporal saliency. Our definition is also a generalization of the temporal saliency technique based on nearest neighbor used in [26].

In this chapter, we present a foreground detection algorithm that computes temporal saliency maps on multiple patch-based features, including two color features, a texture feature, and two types of smooth regions. Saliency maps of color and texture features are combined to increase the signal-to-noise ratio. A simple segmentation algorithm based on adaptive thresholding is applied to the saliency maps to generate candidate foreground maps. A final foreground map is generated by combining the foreground maps of each feature channel.

4.2 Foreground detection algorithm

There are five steps in our algorithm: (1) feature extraction; (2) temporal saliency analysis; (3) feature combination; (4) segmentation; and (5) feedback. A visual representation of our algorithm is shown in Figure 4.1.

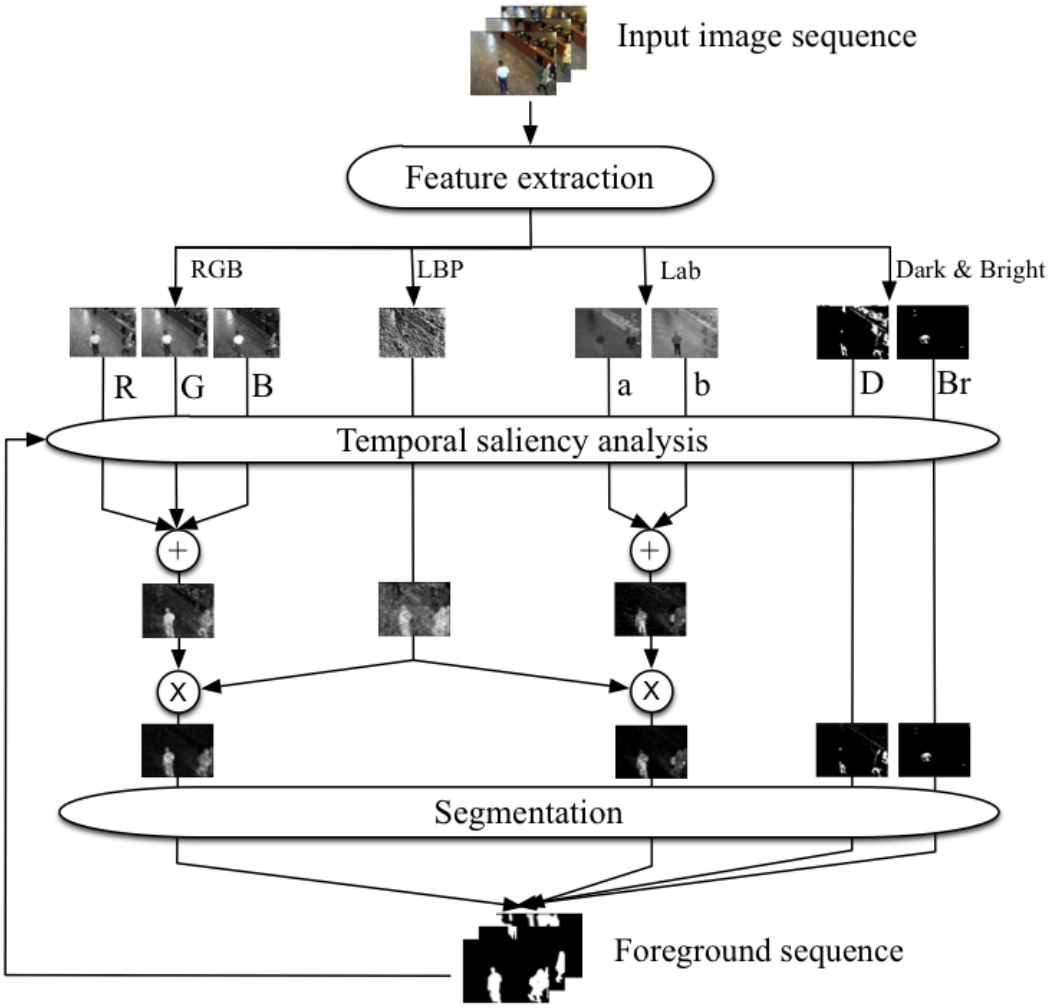


Figure 4.1: The framework of the proposed foreground detection algorithm.

4.2.1 Feature extraction

Like many saliency detection algorithms, we compute temporal saliency on multiple feature spaces. We convert each input frame to five feature spaces: (1) RGB; (2) Lab; (3) LBP; (4) Dark; and (5) Bright. Using both RGB and Lab color spaces achieves better performance than using only one sin-

gle color space [4]. We also discard the lightness channel in the Lab color space when doing temporal saliency. This is because the lightness channel is particularly sensitive to shadows which often lead to false alarms during foreground detection.

In order to further decrease the effect of shadow, we combine the saliency of RGB and Lab color spaces with Local Binary Patterns (LBP) for its property of grayscale invariance [28]. We construct LBP descriptors of each pixel by comparing its grayscale intensity against its eight neighbors:

$$LBP(x_c, y_c) = \sum_{P=0}^{7} s(g_p - g_c)2^p, \qquad s(x) = \begin{cases} 1 & x \geq t \\ 0 & x < t \end{cases} \tag{4.1}$$

where g_c and g_p are the gray values of the center pixel and p_{th} pixel in the 3×3 neighborhood. t is a nonzero threshold for the gray value comparison which could be used to adjust the sensitivity of the LBP operator. LBP is more sensitive to intensity gradients at smaller t. LBP descriptors with a larger t are more robust to noise but are also less distinctive. t is between 0 and 3 in our experiments.

Unfortunately, all of the smooth regions of an image are coded exactly the same by LBP. Therefore, smooth regions of the foreground often have lower saliency values in the LBP channel. This problem happens frequently in the dark and bright regions where texture is very weak. We avoid this problem by including dark and bright features. Dark and bright feature maps are generated by thresholding the input images:

$$D(x, y) = \begin{cases} 0 & I(x, y) \geq t_D \\ 1 & I(x, y) < t_D \end{cases}, \qquad B(x, y) = \begin{cases} 1 & I(x, y) \geq t_{Br} \\ 0 & I(x, y) < t_{Br} \end{cases} \tag{4.2}$$

where $I(x, y)$ is the intensity of the input image and t_D and t_{Br} are the threshold of dark and bright pixels.

After converting input image to all the feature spaces, patch-based feature are computed on all eight feature maps:

$$\mathbf{h}_{g,t}(x, y) = hist\left[W_{g,t}(x, y)\right], \qquad g \in \{R, G, B, a, b, LBP, D, Br\} \tag{4.3}$$

where $\mathbf{h}(x, y)$ and $W(x, y)$ are the histogram and the patch with the center at location (x, y) respectively. t is the temporal index of each image frame.

4.2.2 Temporal temporal saliency analysis

We compute the temporal center-surround difference using the distance of the k^{th} neighbor. k is a tuning parameter that depends on the length and dynamic of the video. The L_1 distance between two histograms is used. We formulate the temporal saliency S as:

$$\begin{aligned} d_g(x,y,t,n) &= ||\mathbf{h}_{g,t}(x,y) - \mathbf{h}_{g,n}(x,y)||_1 \\ D_{g,x,y,t} &= \{d_g(x,y,t,n)|n = 1,...,L\} \\ \mathbf{d}_{g,sort}(x,y,t,j) &= sort\{D_{g,x,y,t}\} \\ S_g(x,y,t) &= d_{g,sort}(x,y,t,k) \end{aligned} \tag{4.4}$$

where $d_g(x,y,t,n)$ is the distance between histograms located at (x,y) of frame t and frame n. g is the same feature index as in Eq. (4.3). $D_{g,x,y,t}$ is the set of distances. L is the number of co-located background image patches. $\mathbf{d}_{g,sort}(x,y,t,j)$ is the sorted distance in ascending order. $S_g(x,y)$ is the magnitude of the temporal saliency. After temporal saliency analysis, we have six maps of temporal saliency.

Saliency based on the distance of the k^{th} nearest neighbor provides a good estimation of the likelihood of foreground activity in videos having highly dynamic scenes.

4.2.3 Feature combination

We combine the saliency maps of the same color space by averaging:

$$\begin{aligned} S_{RGB}(x,y,t) &= (S_R(x,y,t) + S_G(x,y,t) + S_B(x,y,t))/3 \\ S_{ab}(x,y,t) &= (S_a(x,y,t) + S_b(x,y,t))/2 \end{aligned} \tag{4.5}$$

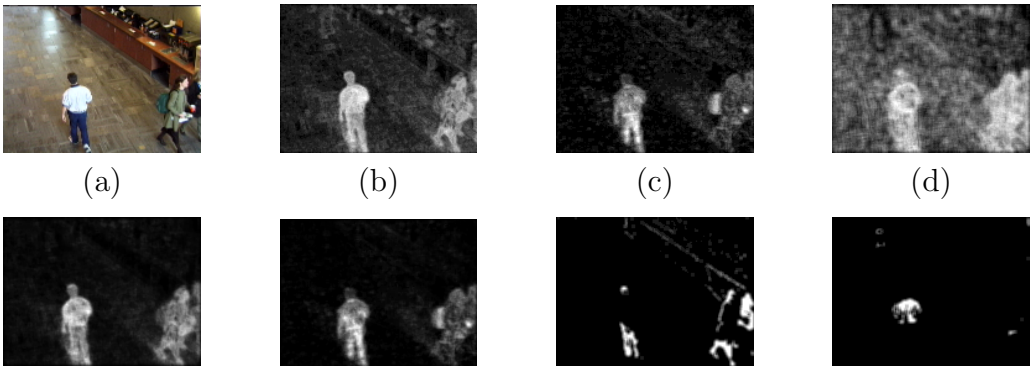
where S_{RGB} and S_{ab} are the temporal saliency map of the two color spaces. The temporal saliency of the two color spaces and LBP is sensitive to noise (Figure 4.2 (b) (c)). This problem can be alleviated by combining the saliency maps of color and LBP. The saliency score of each feature can be treated as the likelihood of foreground. The magnitude of the saliency map of each pixel can be treated as its likelihood of belonging to foreground. Assuming the LBP and two color channels comply with the naïve Bayes assumption,

we have:

$$S_{RGB\_LBP}(x,y,t) = S_{RGB}(x,y,t) \cdot S_{LBP}(x,y,t)$$
$$S_{ab\_LBP}(x,y,t) = S_{ab}(x,y,t) \cdot S_{LBP}(x,y,t) \tag{4.6}$$

where $P(S_{I,x,y,t}, S_{LBP,x,y,t}|\text{foreground})$ is the joint likelihood of pixel (x,y,t) belonging to foreground. Comparing Figure 4.2 (a) with (b), we can observe saliency of the same pixel of the two color features could be significantly different. For example, the green backpack of the woman has much stronger response in S_{ab} than in S_{RGB}. In Figures 4.2 (b)-(f), we can see that there is much less noise in S_{RGB_LBP} and S_{ab_LBP} than in individual saliency maps.

Figures 4.2 (g) and (h) are saliency maps of the dark and bright channels. We can observe that dark and bright regions tend to have strong intensity in these two saliency maps. Because these two features are designed to compensate the drawback of the LBP feature, the saliency of these two features do not combine with the saliency of LBP.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 4.2: Input image and saliency maps: (a) input image; (b) S_{RGB}; (c) S_{ab}; (d) S_{LBP}; (e) S_{RGB_LBP}; (f) S_{ab_LBP}; (g) S_D; and (h) S_{Br}.

4.2.4 Segmentation and feedback

A simple segmentation algorithm is applied to the likelihood map to detect foreground objects. All the pixels are classified as foreground or background by thresholding S_{RGB_LBP}, S_{ab_LBP},S_D, and S_{Br} respectively. An adaptive threshold l is defined by:

$$l = max(l_O, l_m) \tag{4.7}$$

where l_O is an adaptive threshold computed by Otsu's method [29] on the video frame. l_m is a minimum saliency value fixed for all the video frames, and it can be used to avoid misclassifying pixels with weak saliency as foreground when there are no foreground objects. Four binary maps are generated by thresholding and they are combined by a union operation. We then use the connected-components algorithm [30] to label the final binary map. Simple morphological operations are used to discard tiny objects and fill the holes of the foreground objects. A candidate foreground map is generated now. If we feed this information back to the step of temporal saliency analysis, it can help the algorithm model the foreground better. In our experiments, we ignore the foreground pixels when computing the k^{th} nearest distance in the feedback iteration and achieve better accuracy.

4.3 Experiments

4.3.1 Qualitative evaluations

We evaluated the proposed method on multiple datasets. These datasets can be divided into two categories based on their source: (1) new image sequences of wildlife from camera traps [31]; (2) a combination of nine complex scenes from ACMMM03 [32, 33]. The measurement we used to quantify the accuracy is the F-score which is the harmonic mean of their precision and recall:

$$F = \frac{2 \cdot recall \cdot precision}{recall + precision} = \frac{2TP}{2TP + FN + FP} \tag{4.8}$$

where TP, FN, FP are true positives, false positives, and false negatives respectively.

Wildlife datasets Over 1 million images of wildlife species were captured using camera traps in this dataset. Images are in both daytime color and nighttime infrared formats (Fig. 4.3). This dataset is very challenging because it contains scenes that are highly dynamic and cluttered. The highly dynamic nature of these scenes is mainly caused by three factors: (1) low temporal sampling rate; (2) background motion; and (3) significant illumination variations. A subset of the images is manually labeled for evaluating the algorithm and will be made available for public use.

The algorithm is evaluated by performing empty frame detection on the wildlife image sequences. The empty frames are labeled based on the following definition: An image is an empty frame if it has no visible region belonging to an animal. True positives are true empty frames.

The evaluation dataset has 100 sequences (1277 images with 265 images as empty frames). The sequences were captured from various locations and, as a result, depict very different scene configurations. There are twenty species and each species has five different image sequences. These species have diverse sizes and behaviors. The sampling rate is lower than three frames per second and the length of each sequence is between five to forty frames.

In this experiment, the distance of the second nearest neighbor ($k = 2$) is used to compute temporal saliency. To detect empty frames, our algorithm computes a score of the nonempty frame for each video frame based on the spatial saliency of the main foreground segment which has the strongest temporal saliency. In each foreground likelihood map, we pick the foreground segment with the largest average intensity and compute its normalized histogram. We also compute the normalized histogram of the background region. The Bhattacharyya distance between these two histograms is used as the spatial saliency score. For each video frame, if this score is smaller than a fixed threshold, it is predicted as an empty frame.

Table 4.1 shows the accuracy of the empty frame detection on the dataset in comparison with (1) RPCA+OF: the Robust PCA plus optical flow approach [14] and (2) EVOC: the video object graph cut approach [26]. It can be seen that the proposed method significantly outperforms the other two methods in this dataset. Figure 4.3 shows two examples of nonempty video frames. The first row is the input image with the enlarged foreground animal on the top-left corner. The second row is the output image with the detected animal in the red bounding box.

ACMMM03 datasets These datasets contain nine videos of different dynamic scenes. These videos have several challenging properties: dynamic background of indoor and outdoor scenes, indoor busy scenes with a moving shadow, and light switching. In these datasets true positives are true foreground pixels.

Table 4.2 shows the accuracy of empty frame detection on this dataset in comparison with (1) PKDE: the pattern kernel density estimation method [32]

and (2) EVOC: the ensemble video object cut approach [26]. It can be seen that the proposed method has the highest accuracy on most test sequences. Figure 4.4 shows four examples of foreground detection and the segmentation result using the proposed method. The first row is the input video frame. The second row is the output image with the detected foreground objects.

(a) (b)

Figure 4.3: Examples of foreground detection on the camera trap dataset. The first row is the original video frame with the enlarged animal on the top-left corner. The second row is the detection results. (a) Bird and (b) Mouse.

Table 4.1: Performance comparison of empty frame detection on camera trap dataset.

	RPCA+OF [14]	EVOC [26]	This work
Precision	0.4123	0.6917	**0.8060**
Recall	0.6830	0.6776	**0.8151**
F score	0.5142	0.6845	**0.8105**

(a) (b) (c) (d)

Figure 4.4: Examples of foreground detection on ACMMM03 dataset. (a) AirportHall, (b) Lobby, (c) Curtain, and (d) WaterSurface.

Table 4.2: Performance comparison of F-scores (%) on the ACMMM03 video sequences.

Sequences	$\text{PKDE}^{w=1+2+3}_{mb-siltp}$	EVOC	This work
Bootstrap	72.90	76.14	**79.89**
AirportHall	68.02	81.65	77.94
Curtain	92.40	93.63	**95.23**
Escalator	68.66	68.24	**78.52**
Fountain	85.04	84.00	**85.46**
ShoppingMall	79.65	79.46	**82.42**
Lobby	79.21	83.86	82.50
Trees	67.83	89.12	**89.40**
WaterSurface	83.15	94.95	93.13
Average	78.69	83.45	**84.89**

CHAPTER 5

SALIENCY IN AUDIO: SALIENCY-MAXIMIZED AUDIO VISUALIZATION

Browsing large audio archives is challenging because of the limitations of human audition and attention. However, this task becomes easier with a suitable visualization of the audio signal, such as a spectrogram transformed to make unusual audio events salient. This transformation maximizes the mutual information between an isolated event's spectrogram and an estimate of how salient the event appears in its surrounding context. When such spectrograms are computed and displayed with fluid zooming over many temporal orders of magnitude, sparse events in long audio recordings can be detected more quickly and more easily. In particular, in a 1/10-real-time acoustic event detection task, subjects who were shown saliency-maximized rather than conventional spectrograms performed significantly better. Saliency maximization also improves the mutual information between the ground truth of non-background sounds and visual saliency, more than other common enhancements to visualization.

5.1 Introduction

Computers with gigabytes of storage have recently become inexpensive enough to be devoted to the routine task of recording audio, producing recordings far longer than those from the entertainment industry. Purely acoustic applications of long recordings include intelligence, surveillance and reconnaissance (ISR), and tracking wildlife such as songbirds and whales [34]. Equally well, such computers can record nonacoustic time series scaled into the human-audible frequence range, from sensors as diverse as accelerometers, EEGs, voltage spike detectors, and seismometers.

Derivation of insights from such recordings cannot be entirely formalized. Purely automatic analysis is inaccurate enough to justify putting a human

investigator in the loop. Thus, we describe the initial stage of investigation as *browsing:* getting a rough feel for the data, for distinguishing its conventional background from surprising outliers. However, online publication of such long recordings has been discouraged by a lack of software for browsing them. Conversely, the lack of published recordings has similarly discouraged the development of audio browsers. This vicious circle is broken by the visualizations described in this chapter, optimized for the human investigator's senses, which enable rapid browsing of even multi-day recordings.

These visualizations are implementable as lightweight browser plugins or WebGL applications. These would allow the uploading of unannotated long recordings with sparse moments of interest, such as recordings of a whale migration or the 18-day demonstration in Tahrir Square, with confidence that those moments can be found rapidly by others.

Machine perception is often outperformed by human perception, as the latter better handles the semantic gap between noisy observations and target events. In the particular field of acoustic event detection (AED), machines poorly detect and label non-speech events in long recordings. For example, none of the systems competing in the CLEAR 2007 AED competition exceeded 30% accuracy in labeling events like quiet chair squeaks in seminar-room recordings [35, 36, 37]. It is similarly easy to devise AED tasks where trained human listeners strongly outperform machines. Here are three examples: humans can detect rifle magazine insertion clicks with 100% accuracy at 0 dB SNR in both white noise and jungle noise [38]; they can count cough events from audio with an inter-transcriber RMS error of less than 4% [39]; and they can detect anomalous events in musical recordings in a single real-time audition [40].

Unfortunately, the power of human audition is constrained by time. For example, most people cannot comprehend continuous speech faster than twice normal speed [41]. At high speed, non-speech acoustic events are even harder to perceive than speech, because most interesting non-speech events are transient and thus disproportionately masked. Worse yet, even after detecting an event in a long segment, pinpointing the event's timestamp usually requires rewinding and replaying. Our experiments show that AED by pure listening is considerably slower than real-time playback.

This real-time barrier to human AED can be broken by enlisting human vision, which efficiently interprets complex scenes at a glance [42, 43, 44]. To

this end, we propose a visualization, a saliency-optimized audio spectrogram in which background audio with unchanging texture is dimmed, to enhance the at-a-glance salience of anomalous audio. This modified spectrogram can be rapidly skimmed using our high-speed zooming software [45], to rapidly identify sparse interesting intervals. This visualization is synchronized with the source recording, so an analyst searching for target events typically listens to only brief excerpts of visually interesting segments. The target events' information is embedded into visually salient patterns, which are processed by human vision with priority [44]. In other words, this visualization suppresses uninteresting background noise.

We formulate this visualization problem as maximizing the mutual information (MI) [46] between the spectrogram of a target event Y and the estimated visual saliency of the examined spectrogram $\varphi(f)$ (Figure 5.1). The input information Y is the spectrogram of the target event in isolation (without the background noise N); the transmitted information is the observer's visual perception. The visualization function f converts the mixed-signal spectrogram X to the saliency-optimized spectrogram. The saliency map $\varphi(f)$ is the output of the saliency model, which models the human visual system's signal-driven (bottom-up) allocation of attention. The visualization $f(X)$ maximizes the MI $I(Y; \varphi(f(X)))$ between noise-free events and the saliency map of training examples. To evaluate $f(X)$, we use it to generate images from another set of target events, disjoint from those used to train $f(X)$. These images are presented to human subjects, whose task performance we then measure.

5.2 Saliency-maximized audio visualization

Let the spectrogram of the target acoustic event be $Y[n_1, n_2]$, an RGB matrix indexed by row index n_1 and column index n_2, where the time scale of the row and column index depend on the zoom of the display (Figure 5.1). In an AED task, users do not observe $Y[n_1, n_2]$ directly; instead, they observe $X[n_1, n_2]$, the spectrogram of the signal mixed with background noise. The background noise, with spectrogram $N[n_1, n_2]$, is irrelevant to the task (e.g., orchestral music [40] or speech [47]). To help users correctly identify where $Y[n_1, n_2]$ is nonzero, we propose to transform the image prior to display,

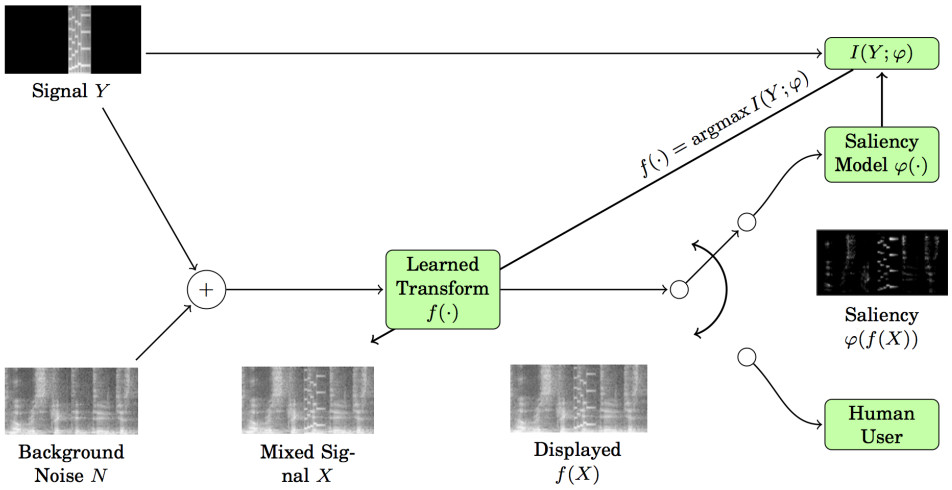


Figure 5.1: Flowchart of human acoustic event detection from a visual display. X is a spectrogram of the input mixed audio signal summing the audio waveform of Y and N, from which the system computes the displayed image $f(X)$. The transformation from spectrogram to displayed image is learned in order to optimize the mutual information $I(Y;\varphi(f(X)))$, where $\varphi(\cdot)$ is a model of human bottom-up attention allocation (saliency). After learning, the transformed image $f(X)$ is displayed to human users to speed up their search for anomalies.

using a learned image transformation $f[n_1, n_2] = f(X[n_1, n_2])$.

The parameters of the learned transformation f maximize how much information about the target event's location is communicated to the user. The information communicated through a noisy channel can be measured as

$$I(Y;\Phi) = E_{Y,\Phi}\left[\log_2\left(\frac{p_{Y,\Phi}(y,\varphi)}{p_Y(y)p_\Phi(\varphi)}\right)\right] \tag{5.1}$$

where Y is the input of the channel, Φ is the output of the channel, and $p_{Y,\Phi}(y,\varphi)$ is their joint probability density with marginals $p_Y(y)$ and $p_\Phi(\varphi)$. Notice that $I(Y;\Phi)$ is an expectation. Because the signals used to train the system differ from those shown to the user during evaluation, our algorithms are therefore trained by maximizing the stochastic approximation

$$\hat{I}(Y;\Phi) = \sum_{t=1}^{T}\log_2\left(\frac{\hat{p}_{Y,\Phi}(y_t,\varphi_t)}{\hat{p}_Y(y_t)\hat{p}_\Phi(\varphi_t)}\right) \tag{5.2}$$

where (y_t, φ_t) are input-output pairs observed in the training corpus, and $\hat{p}_{Y,\Phi}(y_t, \varphi_t)$ is the binned empirical probability mass function of the training data.

The channel output Φ deserves further comment. Our visualization application communicates with listeners through a channel that includes four types of distortion: (1) the addition of background noise, (2) the scaling of the spectrogram, modeled by choosing the appropriate time scale for indices n_1 and n_2, (3) the intentional distortion caused by the learned mapping $f(X[n_1, n_2])$, and (4) the limited processing power of human visual attention. We approximate the human visual system with a communication channel that attends to visual patterns selectively, in decreasing order of saliency. Therefore, when quickly examining a display, it perceives at most a few highly salient objects. The rate of information transmission is limited by the finite span of attention (about six objects at a glance), and by immediate memory (about seven items) [48]. Our model of the communication channel is a sort of homunculus model, according to which a high-level cognitive process detects exactly those acoustic events whose visible evidence is attended to by the low-level visual attention system. The image received by the high-level cognitive processes is therefore $\varphi[n_1, n_2] = \varphi(f[n_1, n_2])$, where $\varphi(f[n_1, n_2])$ is a nonlinear multi-scale saliency transform, based on a saliency model that ze-

ros out all pixels of $f[n_1, n_2]$ except for those that will likely attract attention (the "salient" pixels) [7].

Saliency of information is important in the field of human factors engineering [49]. Examples of this include a quality metric for visualization that uses the correspondence between a data relevance mask and a saliency map [50], and a saliency-based perceptual tool for visual design [51]. But although saliency has been used to *analyze* the quality of a visual representation, it has not yet been effectively used to automatically *generate* saliency-maximized visual representations.

We propose to measure the efficiency of information transmission from Y to φ through their MI. The visualization (encoding) function f is chosen to maximize $I(Y;\varphi)$, to represent the target events optimally for fast human visual examination. Hence,

$$f^* = \underset{f}{\operatorname{argmax}}\, \hat{I}(Y; \varphi(f(X))) \tag{5.3}$$

where X is the input spectrogram, Y is the ground truth (the spectrogram of the isolated event), and $f(X)$ is the displayed spectrogram, a transformation of X. Five modules solve for the optimized transformation function f: computing the spectrogram, transforming the visualization, computing the saliency map, computing the MI, and maximizing the MI (Figure 5.1). Optimization of f uses only the training data; entirely separate acoustic events and background noise are used to evaluate the derived f^*.

5.2.1 Computing and transforming the spectrogram

We base our visualization on the humble spectrogram because it is familiar to audio experts, and because even naïve subjects can successfully interpret its details. Our grayscale spectrogram resolves 128 frequency bands down to 5 msec, in linear rather than logarithmic frequency scale so as to preserve the high-frequency information that is useful for distinguishing target events.

Our goal is to find a transformation function f that is saliency-maximized, rendering target events so that φ extracts them as salient patterns. For simplicity we use linear filters: $f(X) = h[n_1, n_2] \ast\ast X[n_1, n_2]$, where $\ast\ast$ denotes 2D convolution, and Eq. (5.3) optimizes h.

5.2.2 Computing the visual saliency map

The saliency map is generated by an image saliency algorithm based on prior work [7, 52], estimating how the human visual system allocates attention to any particular pixel [53, 54]. Our saliency algorithm is independent of acoustic events, with parameters chosen empirically from the literature. During training, we used this algorithm to learn a spectrogram transformation function, the linear filter h in Section 5.2.1. It was used as well in objective evaluation, to evaluate competing visualizations. Of course it was not needed in subjective evaluation, which used actual humans instead of algorithmic estimation. Our algorithm has three steps: building feature pyramids; computing each feature's center-surround difference; and combining all features' saliency maps into a single map (Figure 5.2).

An important step in any biologically plausible model of bottom-up attention is the parallel computation of early visual features such as intensity, orientation and color. Because we use a gray-scale spectrogram, we omit color, leaving orientation and intensity [7]. These features are computed by filtering the displayed image: $I_k[n_1, n_2] = B_k[n_1, n_2] **f[n_1, n_2]$, where $k \in \{I, 0°, 45°, 90°, 135°\}$. The filter $B_I[n_1, n_2] = \delta[n_1, n_2]$ is the delta function (so $I_I = f$). The other four filters are Gabor filters with orientations of $\{0°, 45°, 90°, 135°\}$. A strong response of $[n_1, n_2]$ in I_k indicates that f has property k in a neighborhood of $[n_1, n_2]$. For example, $I_{45°}[n_i, n_j]$ responds strongly to a 45° bar at $[n_i, n_j]$ in f.

Because a salient region differs from its neighborhood, the algorithm detects saliency with a center-surround difference (CSD), implemented by convolving the input image with a difference of Gaussians (DoG) filter. Thus $\mathrm{CSD}_k[n_1, n_2] = I_k[n_1, n_2] **\mathrm{DoG}[n_1, n_2]$, where

$$\mathrm{DoG}[n_1, n_2] = \frac{1}{2\pi}\left(\frac{e^{-(n_1^2+n_2^2)/2\sigma_c^2}}{\sigma_c^2} - \frac{e^{-(n_1^2+n_2^2)/2\sigma_s^2}}{\sigma_s^2}\right) \tag{5.4}$$

This DoG function is parameterized by two σ's. The first Gaussian has the smaller σ_c; the second has the larger σ_s. (Subtracting these Gaussians approximates a Mexican hat function.) Filtering an image with the central Gaussian, the first term in Eq. (5.4), averages the features within approximately σ_c pixels of the output pixel. The surround Gaussian, the second term, computes a similar average for σ_s. Thus $\mathrm{CSD}_k[n_1, n_2]$ estimates how

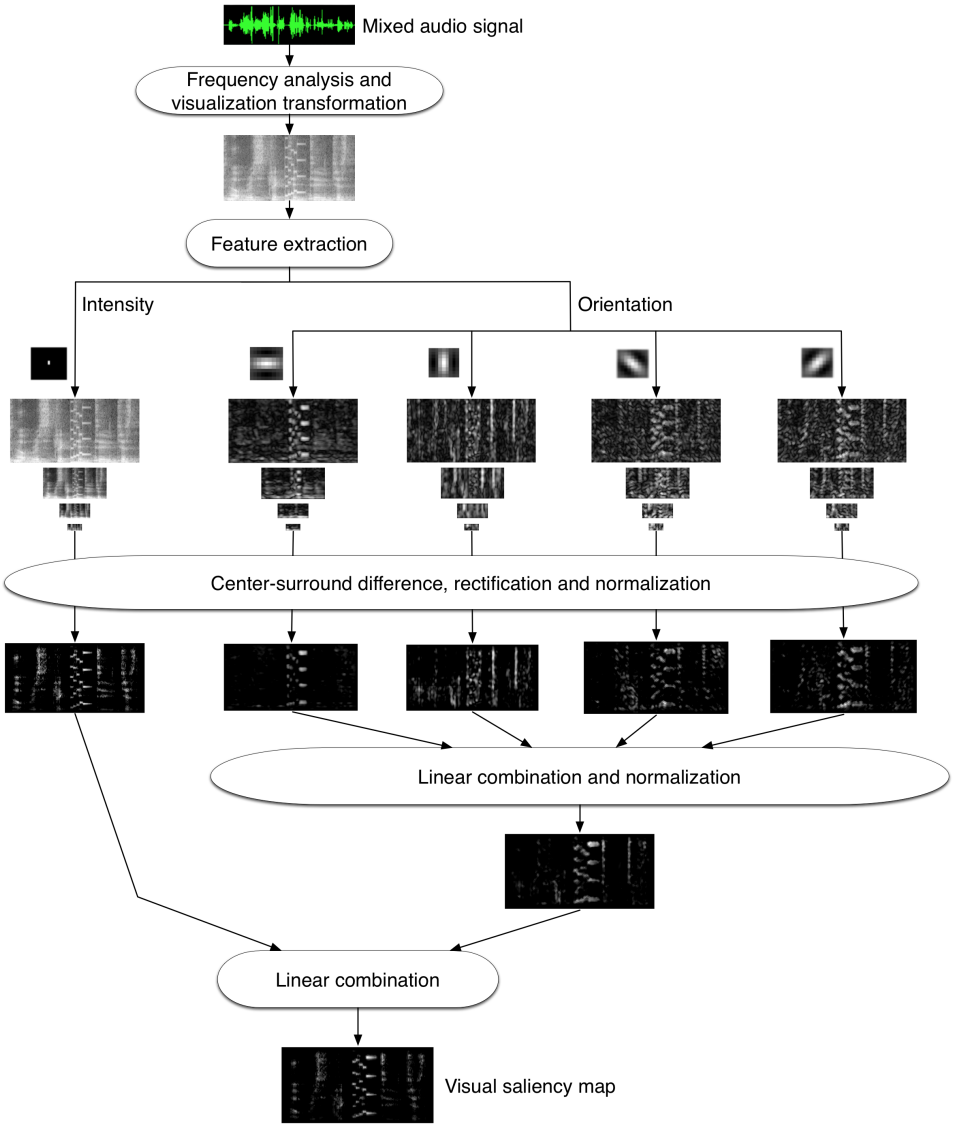


Figure 5.2: Computation of visual saliency. The mixed audio signal is converted to a spectrogram and then transformed to a displayed image. (Here the transformation function is the initial transformation of $\delta[n_1, n_2]$ in Section 5.2.3, so the image is the spectrogram itself.) From this, a delta-function filter and four truncated 9×9 Gabor filters (enlarged here for clarity) extract intensity and orientation features. Center-surround difference (CSD) is implemented by across-scale subtraction between layers of a Gaussian pyramid. To detect strong centers on weak surrounds, each CSD is halfwave rectified. The CSDs are then normalized and combined, yielding a single saliency map.

much the pixels near (n_1, n_2) stand out, relative to those in the surrounding disc of radius σ_s.

Because Gaussian filtering with large σ's is computationally expensive, we approximate it by filtering recursively with a small Gaussian kernel and downsampling. A dyadic Gaussian pyramid generates images filtered by Gaussians of different σ's. The displayed input image is filtered by a 2D separable Gaussian kernel $[1\,5\,10\,10\,5\,1]/32$ and downsampled twofold. Repeating this builds the layers of the pyramid [52]. The filters B_k are applied to the Gaussian pyramid to extract features from different layers:

$$\mathrm{CSD}_k = \max\left\{0, F_{k,c} \ominus F_{k,s}\right\}, \qquad k \in \{I, 0^\circ, 45^\circ, 90^\circ, 135^\circ\} \tag{5.5}$$

where $F_{k,c}$ and $F_{k,s}$ are the center and surround layers of the pyramid for feature k, and $\ominus$ denotes across-scale subtraction. We use the pyramid's first and fourth layers as center and surround. Fullwave rectification commonly computes the magnitude of the response of weak centers on strong surrounds (or vice versa). We instead use (positive) halfwave rectification in Eq. 5.5 because, in an AED task, target events almost always have more energy than their background.

We combine the CSDs of different features into a single saliency map, normalizing before every summation with a nonlinear operator $N(\cdot)$. This operator restores similar dynamic range to the CSDs of different visual modalities, and also enhances strong local peak response. It is implemented with a large 2D DoG filter. This is followed by positive rectification [11].

The final saliency map $S[n_1, n_2]$ is the mean of the normalized maps for intensity and for the combined orientations:

$$\begin{aligned} \overline{F_I} &= N\left(\mathrm{CSD}_I\right) \\ \overline{F_O} &= N\left(\sum\nolimits_k N\left(\mathrm{CSD}_k\right)\right), \qquad k \in \{0^\circ, 45^\circ, 90^\circ, 135^\circ\} \\ S &= (\overline{F_I} + \overline{F_O})/2 \end{aligned}$$

5.2.3 Maximizing mutual information

To evaluate how well human visual perception captures the information in the visualization associated with the target events, we estimate the MI between

the ground truth Y (the spectrogram of the isolated target event, obtained according to Section 5.2.1) and the saliency map Φ of the transformed spectrogram of the mixed signals (Eq. 5.2).

Because the objective function $\hat{I}(Y;\varphi)$ (Eq. 5.2) is non-convex and non-differentiable, we can only approximate the global maximum. Simulated annealing estimates f from an initial transformation of $h[n_1, n_2] = \delta[n_1, n_2]$. This transformation is also the baseline one, and corresponds to the conventional spectrogram $f[n_1, n_2] = X[n_1, n_2]$.

The audio targets and background noises (y_t, φ_t) used to maximize MI are similar to, but disjoint from, those used in evaluation.

We evaluated linear filters with sizes from 5×5 to 15×15, all with similar optimized mean MI's. For human-subject experiments we chose a 5×5 filter, after inspecting the visualizations generated from the training data.

5.3 Alternative enhancements of visualization

Several traditional algorithms exist for enhancing visualizations. We compare the performance of the proposed algorithm with some popular methods. Figure 5.3 shows the corresponding spectrograms. We tested these enhancements:

(a) Energy thresholding with MI maximization
Thresholding, ubiquitous in human perception [44], suppresses any signal below a threshold. It has been applied in computer vision algorithms such as foreground detection [55, 10]. As an alternative to our proposed saliency map generation, for learning the 5×5 linear visualization transformation filter we replaced saliency map generation with energy thresholding [56], zeroing any pixels with intensity below the threshold.

(b) Event-specific Wiener filter
The Wiener filter suppresses noise using a filter that minimizes mean-squared error using the known autocorrelations and cross-correlations of the input signals [57]. We gave each evaluation event its own Wiener filter, unrealistically using knowledge of the autocorrelations and cross-correlations of the noisy signal and target event of each testing input

spectrogram. This oracle let us investigate the performance upper bound for audio visualization using the Wiener filter.

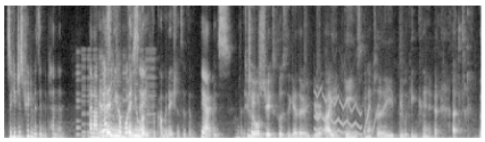
(c) Event-independent Wiener filter
Realistic audio browsing lacks event-specific autocorrelations and cross-correlations. Here, we estimated one filter for all target events, using the average autocorrelation and cross-correlation of spectrograms in the training corpus. We trained a single 25×25 Wiener filter based on the statistical property of the patches of the training spectrograms. This filter was then applied to the evaluation spectrograms, to estimate how effective Wiener filtering is under real-world browsing conditions.

(d) Non-negative matrix factor deconvolution
Besides noise filtering, we also tested an example of blind source separation, namely non-negative matrix factor deconvolution (NMFD), an extension of non-negative matrix factorization (NMF) [58, 59]. NMF has been applied to magnitude spectrograms of polyphonic music, modeling each instrument with an instantaneous frequency signature [60]. Extending this, NMFD models each instrument with a time-frequency signature by considering temporal structure [61]. NMFD has two important parameters, the number of basis functions R and the temporal length of the factors T. We tuned these parameters for the training set, and applied them to the evaluation set (we used $R = 17$ and $T = 31$). Some of the basis functions were chosen to reconstruct the target events. There were multiple combinations of basis functions for partial reconstruction of the spectrogram. We chose the reconstructed spectrogram with the highest correlation coefficient to the ground truth event, again as an oracle for an upper bound on NMFD's performance.

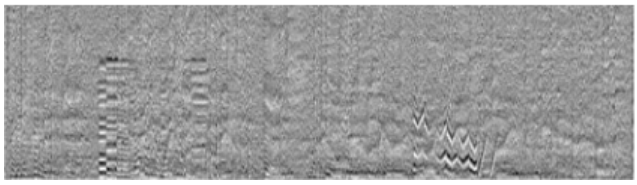
5.4 Objective evaluation

Data for training and evaluating the algorithm used electronic sound effects, such as those common to 1980s video games, as target events, superimposed on the realistically noisy background of an ongoing seminar [62]. All 62 sound effects were obviously foreign to a seminar room. The lengths of the sound

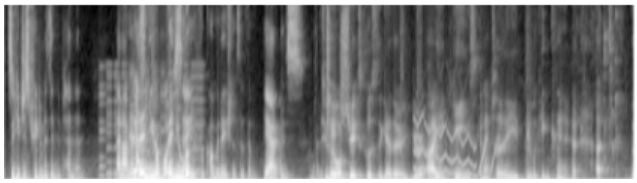
(a) Spectrogram of two target events in isolation.



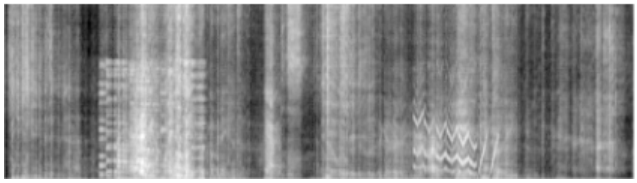
(b) Spectrogram of the same events, with background noise.



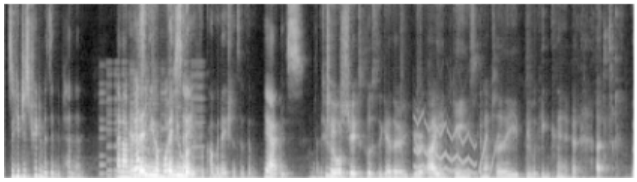
(c) Saliency-maximized spectrogram (our proposed method).



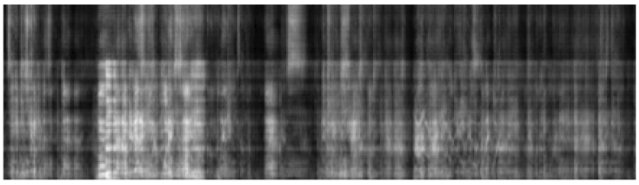
(d) Spectrogram enhanced with maximized MI, based on energy thresholding instead of visual saliency.



(e) Spectrogram enhanced by event-specific Wiener filtering (an oracle result).



(f) Spectrogram enhanced by event-independent Wiener filtering.



(g) Spectrogram enhanced by non-negative matrix factor deconvolution (upper bound).

Figure 5.3: Various visualizations of spectrograms.

effects are shorter than five seconds except for one lasting for nine seconds. Both the target events and the background audio were split into disjoint subsets, half for training and half for evaluation. Samples for training and evaluation were made by adding each target event to a temporally center-aligned background four times longer than the event itself.

The saliency-maximizing transformation learned by our proposed algorithm emphasizes non-speech events and attenuates the background, which is strongly heterogeneous and has significant speech presence (Figure 5.4). The three target events in the figure are obscured in the conventional spectrogram, but instantly visible in the saliency-maximized spectrogram.

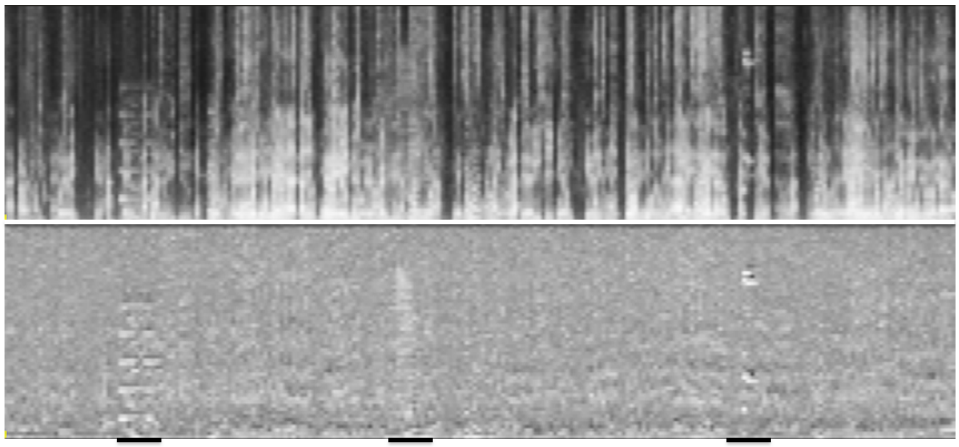


Figure 5.4: Qualitative enhancement of a spectrogram: conventional (top), saliency-maximized (bottom).
Three target events are marked with black underlines.

Our objective measure is the empirical MI between the saliency map of the spectrogram and the ground truth. (Each spectrogram has its own saliency map, generated by the same algorithm.) Figure 5.5 shows the quantitative improvement due to maximizing saliency. Both axes measure the $I(Y;\varphi)$ of evaluation samples. (Recall that neither these samples nor these backgrounds were used in training.)

The proposed algorithm most improves low-SNR target events that are still barely visible in the unenhanced spectrogram. The net improvement of the MI between conventional and saliency-maximized spectrograms of evaluation samples is maximized around -20 dB SNR (Figure 5.6). At very low SNR, the target's visual pattern is too buried to be enhanced. Conversely, at very high SNR the visual pattern is already so salient that little improvement is

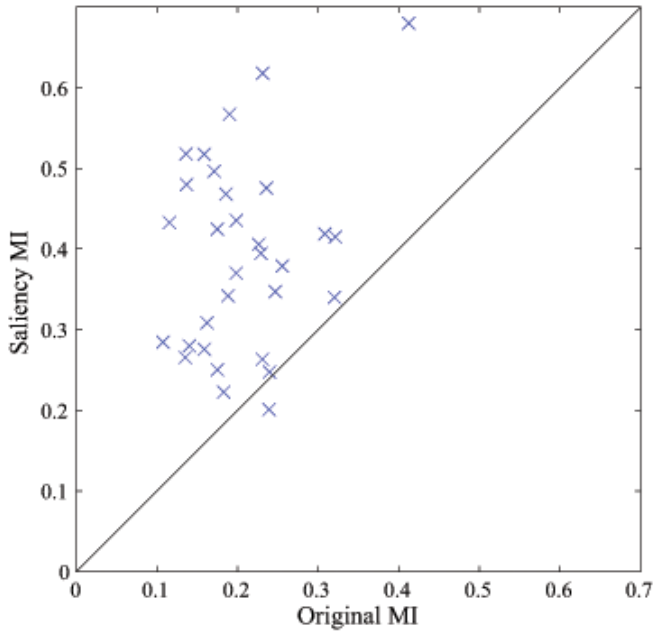


Figure 5.5: Comparison of MI for 31 evaluation samples. Almost all samples yield a larger MI when the spectrogram is saliency-maximized.

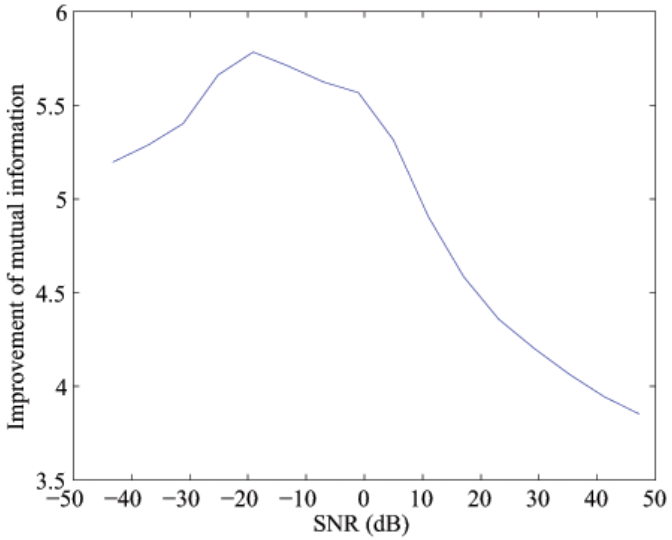


Figure 5.6: Sensitivity to SNR of MI improvement due to saliency maximization.

possible.

We compare the performance of the proposed algorithm with the alternatives discussed in Section 5.3. Figure 5.7 shows that the proposed method had the largest average MI improvement. NMFD and the event-specific Wiener filter were second best, the latter with smaller standard deviation. Compared to baseline, energy thresholding and event-independent Wiener filtering actually degraded performance. Because the proposed method actually uses MI to improve the spectrogram, it may be unfairly favored in the comparison of Figure 5.7. Nevertheless, these results are corroborated by another measure of performance, the correlation coefficient. Figure 5.8 shows that NMFD had the largest average correlation improvement. The proposed method and event-specific Wiener filtering performed similarly, while energy thresholding and event-independent Wiener filtering again perform worse than baseline. This order may be explained because the oracular event-specific Wiener filter uses the most information about the evaluation spectrogram; NMFD also uses correlation with the ground truth to choose the spectrogram's best partial reconstruction. This provides crucial prior knowledge about the target event's temporal location and visual pattern, for choosing the most related components for partial reconstruction. Of the three enhancements trained without such "cheating," energy thresholding and event-independent Wiener filtering performed poorly, while the proposed algorithm performed best. The proposed method, using linear filters, is also among the computationally fastest of these visualizations. In particular, it is four orders of magnitude faster than the NMFD that we used (non-negativity constraints often slow down convergence). Note also that what the proposed method tries to increase is a target event's saliency; except for energy thresholding, the alternatives increase an event's SNR, a different (and perhaps harder) problem.

5.5 Visualization-guided audio browser

The Timeliner audio browser (Figure 5.9) displays the waveform of a long audio recording, labeled in units ranging from weeks down to milliseconds depending on the current zoom level [45]. Timeliner also displays audio visualization features such as audio spectrograms or the outputs of event classifiers. The user can smoothly zoom in to interesting subintervals, to

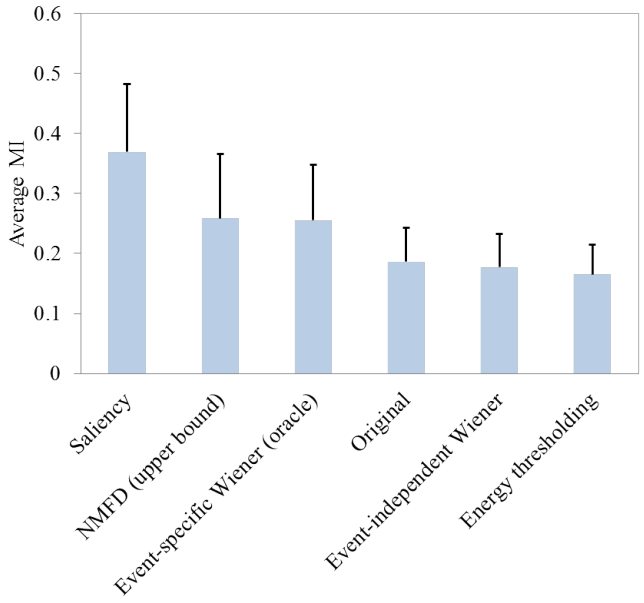


Figure 5.7: Average MI using different methods (error bars indicate standard deviation).

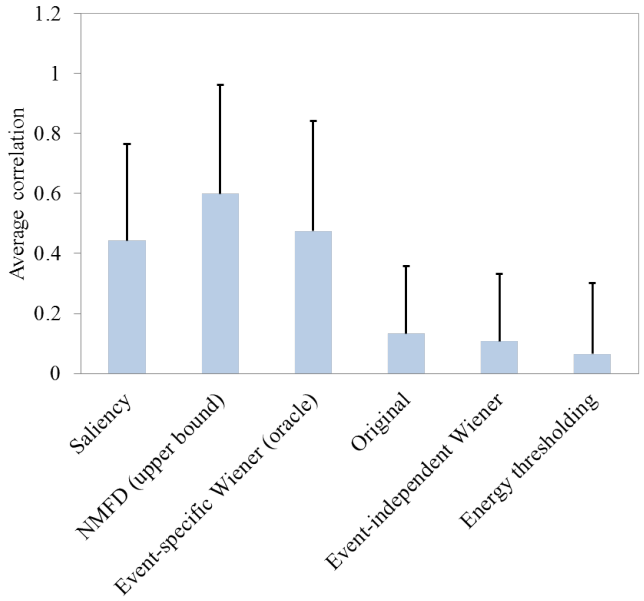


Figure 5.8: Average correlation coefficients using different methods (error bars indicate standard deviation).

find even very brief anomalous segments without long periods of listening.

In conventional audio editors and browsers, the color of each pixel or texel is computed by undersampling data from the corresponding time interval. In Timeliner, such an interval might be tens of minutes long, and naïve undersampling flickers distractingly when panning and zooming at 60 frames per second. This flickering entirely obscures the data until the pan or zoom stops, eliminating any benefit of these continuous gestures.

To restore smoothness and to improve performance for long recordings, Timeliner first computes a multiscale cache for the recording and for the derived features. Given any subinterval of the recording, this cache yields the minimum, mean, and maximum data values found during that interval, for either scalar or vector data. (The time taken to compute this min-mean-max is logarithmic with respect to the full recording's duration, and independent of the subinterval's duration.) Final rendering maps each pixel's or texel's triplet to a hue-saturation-value color, through a predefined transfer function.

Inspired by the left hand on the keyboard / right hand on the mouse layout that emerged in 1990s real-time games, Timeliner's two-handed input frees the user's gaze from hunting for keys. The "WASD" keys pan and zoom, while the spacebar starts and pauses audio playback. The mouse and its scrollwheel also pan and zoom, so users can choose whatever modality they find most familiar.

Timeliner's file parsing and user interface are implemented in the scripting language Ruby, while its heavier computation is done in C++ to reduce memory usage. Graphics are rendered with OpenGL and its utility toolkit GLUT. Timeliner runs natively on Linux and Windows, and is distributed open source.

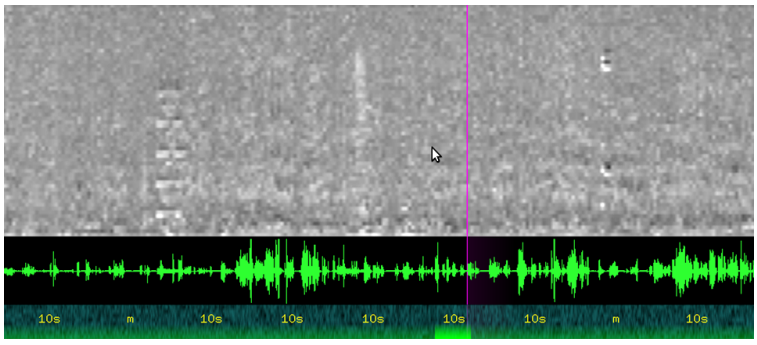


Figure 5.9: Components of Timeliner's interface: saliency-maximized spectrogram, waveform, and time axis.

5.6 Subjective evaluation

We measured human subjects' AED performance with both conventional and saliency-maximized spectrograms, using an otherwise identical computer interface. Timeliner enabled convenient audio browsing. Video was presented with a 17-inch CRT, audio with ear buds (Figure 5.10).

Figure 5.10: Configuration of the human subject experiment.

We asked twelve subjects, unfamiliar with spectrograms, to detect anomalous target events in 80-minute recordings of seminar room background noise [62]. Into each recording we mixed 40 sound effects randomly chosen from the objective evaluation set of 31 different ones, uniformly distributed but without overlap, at various amplitudes. Because the task lasted only 8 minutes, naïve listening (real-time search) would expect to find only a tenth of the targets. We therefore instructed subjects to first scan for a visually suspicious pattern and then verify it by listening, before annotating that target's temporal position.

Each subject annotated six different recordings, using either three saliency-maximized followed by three conventional spectrograms, or the reverse order. This ordering was balanced across subjects. The first and the fourth sessions were just for practice with each visualization; only the other sessions were evaluated for performance. The recordings used in the non-practice sessions were balanced across subjects. To restore subjects' vigilance and to reconfigure the computer between sessions, subjects rested for about one minute, or longer if they desired. (This is also why we did not use one very long session.) Afterward, subjects were asked which spectrogram was more helpful (we ex-

plained nothing to them about spectrograms or saliency). All preferred the saliency-maximized one.

To quantify subjects' AED performance from their annotated timestamps, we computed their recall and their precision. Recall was the fraction of targets whose durations contained a timestamp (how many were hit). Precision was the fraction of timestamps that were in some target (hits per try). A subject's F-score was the harmonic mean of their precision and recall [63].

The experiment was a within-subject design, comparing subjects' F-scores when using either conventional or saliency-maximized spectrograms. A paired samples t-test revealed a significant difference in the F-scores for conventional ($M = 0.2752$, $SD = 0.0777$) and saliency-maximized ($M = 0.5846$, $SD = 0.1033$) spectrograms; $t(11) = -12.976$, $p < .05$. This suggests that the saliency-maximized spectrogram significantly outperformed the conventional one.

Maximizing saliency increased not only F-scores but also stability. Figure 5.11 shows that it increased the number of events found within 30 seconds, from 47% to 74%, and also decreased the longest time between detections, from 5 minutes to 3 minutes. The longer tail for the histogram of the conventional spectrogram correlates with the frustration that many subjects reported while using it.

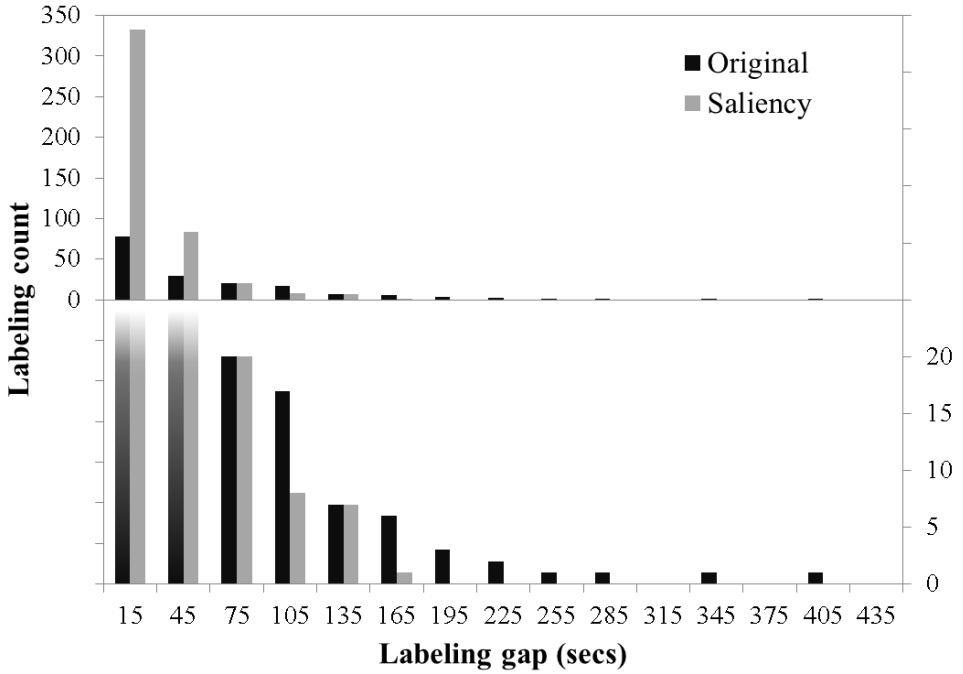


Figure 5.11: Histogram of durations between consecutive correct labelings. The lower subfigure's ordinate is magnified, to better display values at gaps over 75 seconds.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Contributions

In this dissertation, we study saliency and its applications in audio and visual signals. In addition to proposing a novel framework of saliency detection based on local outlierness using the k^{th} nearest neighbor distance, we also look into new applications of saliency on images, image sequences, and audio. We summarize the contributions of this dissertation as follows:

- We propose a novel image saliency algorithm based on outlier detection. Outlierness of a local patch is measured using KNN distance between its center region and its surround subregions. The proposed algorithm is compared with two popular saliency algorithms and achieves better performance.

  Another important advantage of using KNN distance to estimate saliency is that KNN distance could be applied to high-dimensional features. In Chapter 4, KNN distance is used to detect temporal saliency in image sequences. We successively apply temporal saliency to do foreground detection and achieve state-of-the-art results. Note that although the saliency maps used in Chapter 3 and Chapter 5 were generated by DoG, they can be replaced by saliency maps generated by the proposed algorithm using KNN distance.

- We design a novel license plate detection method based on image saliency. Segmentation based on the intensity saliency map help us detect the characters of a license plate precisely with a high recall rate. The following license plate detection with saliency-related features effectively eliminates the potential false positives. Our experiments show that the saliency-related features are important to our algorithm. The ro-

bustness of this method is verified by applying it to a mixed dataset with high diversity. Another advantage of the proposed algorithm is that the detection box has high precision. This property should help character recognition which is important to most applications of license plate detection.

- We developed a novel foreground detection algorithm based on saliency for highly dynamic and cluttered scenes. The distance of the k^{th} nearest neighbor is used to detect temporal saliency in an image sequence. Saliency is computed on patch-based features, including two color features, a texture feature, and two types of smooth regions. These saliency maps are used as likelihood maps to do foreground segmentation using adaptive thresholding. In our experiments on multiple datasets, the proposed algorithm achieves state-of-the-art results.

- Our proposed saliency-maximized spectrogram enables audio browsing that is much faster than real time. In AED it improves the mutual information between the ground truth of non-background sounds and visual saliency, more than other common enhancements to visualization. In a 1/10-real-time AED task, compared to conventional spectrograms, it increased stability and improved subjects' F-score by 100%.

6.2 Future work

In this section, we list several potential research directions.

6.2.1 Top-down features for fixation prediction

The proposed saliency algorithm based on KNN distance has reasonably good performance to predict fixation using low-level features. In order to improve the accuracy of fixation prediction, top-down factors should be considered. Top-down concepts such as faces, text and people play an important role when human freely view scenes.

One method to incorporate top-down concepts in the algorithm of fixation prediction is using high-level features. Deep neural network is promising to provide high-level features of natural images. For example, people applied

deep convolutional neural network to a big image dataset (over 1 million images) and achieved very good classification accuracy [64]. It was found that the intermediate feature layers of the network could extract high-level features such as eyes, tire, legs, and faces [65]. It will be worthwhile to investigate if considering these high-level features can improve the accuracy of fixation prediction.

6.2.2 Motion saliency

The framework of temporal saliency can be applied to motion features to detect motion saliency. Motion saliency could be used to detect abnormal video events. Abnormal event detection gained importance in the research of video surveillance in recent years. It is a very challenging task because of the lack of training data for abnormal events. There are two related topics we are interested in:

1. Saliency in a spatio-temporal window
   In this dissertation we successively use KNN distance to detect saliency in a spatial neighborhood for images and saliency in a temporal neighborhood for image sequences. For motion saliency, the motion pattern is usually described by spatio-temporal patches. The saliency of a center spatio-temporal patch could be computed by KNN distance in its spatio-temporal neighborhood.

2. Motion pattern representation
   Similar to image saliency, good features are important to motion saliency. There are some potential motion features for motion saliency detection. For example, optical flow will be a good candidate for describing low-level motion patterns. A histogram of optical flow could be an option for describing motion patterns of higher level. Besides these two motion features, there are more motion pattern representations need to be explored.

6.2.3 Information visualization

The proposed algorithm of audio visualization could be extended and applied to time series derived from non-acoustic sensors. We also wish to use more

refined models of human perception, such as color saliency, even though these will require more training data.

Finally, we wish to apply this enhancement to more realistic audio events. (It increased the saliency of a few background sounds, notably rapidly rising pitch in female speech and squeaky writing on a whiteboard; these were indeed sometimes mislabeled as anomalous.) A potential application is WhaleFM which analyzing spectrograms of whale's sounds using the collective intelligence of volunteer participants [34]. Scientists spend much time analyzing long audio recordings to extract audio segments of whale sound and ask the participants to cluster the spectrograms of the whale sounds. We found many of the audio samples on WhaleFM sound similar to the sound effects used in our experiments. With our audio visualization system, the efficiency of the whale sound analysis could be improved significantly.

REFERENCES

[1] L. Itti, "Automatic foveation for video compression using a neurobiological model of visual attention," *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1304–1318, 2004.

[2] R. Achantay, S. Hemamiz, F. Estraday, and S. Süsstrunky, "Frequency-tuned salient region detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[3] H. Liu, S. Jiang, Q. Huang, and C. Xu, "A generic virtual content insertion system based on visual attention analysis," in *Proceedings of ACM International Conference on Multimedia*, 2008.

[4] A. Borji and L. Itti, "Exploiting local and global patch rarities for saliency detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[5] B. A. Wandell, *Foundations of Vision*. Sunderland, MA, USA: Sinauer Associates, 1995.

[6] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 185–207, 2013.

[7] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[8] N. D. B. Bruce and J. K. Tsotsos, "Saliency based on information maximization," in *Proceedings of Advances in Neural Information Processing Systems*, 2006.

[9] C. C. Aggarwal, *Outlier Analysis*. New York, NY, USA: Springer, 2013.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Boston, MA, USA: Addison-Wesley Longman, 2001.

[11] L. Itti and C. Koch, "Feature combination strategies for saliency-based visual attention systems," *Journal of Electronic Imaging*, vol. 10, no. 1, pp. 161–169, 2001.

[12] J. Zhang and S. Sclaroff, "Saliency detection: A Boolean map approach," in *Proceedings of IEEE International Conference on Computer Vision*, 2013.

[13] B. W. Tatler, R. J. Baddeley, and I. D. Gilchrist, "Visual correlates of fixation selection: Effects of scale and time." *Vision Research*, vol. 45, no. 5, pp. 643–659, Mar. 2005.

[14] P. Khorrami, J. Wang, and T. S. Huang, "Multiple animal species detection using robust principal component analysis and large displacement optical flow," in *Proceedings of International Conference on Pattern Recognition, Workshop on Visual Observation and Analysis of Animal and Insect Behavior*, 2012.

[15] V. Kamat and S. Ganesan, "An efficient implementation of the Hough transform for detecting vehicle license plates using DSP'S," in *Proceedings of IEEE Real-Time Technology and Applications Symposium*, 1995.

[16] J. Hsieh, S. Yu, and Y. Chen, "Morphology-based license plate detection from complex scenes," in *Proceedings of International Conference on Pattern Recognition*, 2002.

[17] K. Kim, K. Jung, and J. Kim, "Color texture-based object detection: An application to license plate localization," in *Proceedings of International Workshop on Pattern Recognition with Support Vector Machines*, 2002.

[18] L. Dlagnekov, "Car license plate, make, and model recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[19] F. Porikli and T. Kocak, "Robust license plate detection using covariance descriptor in a neural network framework," in *Proceedings of International Conference on Video and Signal Based Surveillance*, 2006.

[20] T. Kadir and M. Brady, "Saliency, scale and image description," *International Journal of Computer Vision*, vol. 45, no. 2, pp. 83–105, 2001.

[21] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of adjacent contour segments for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 1, pp. 36–51, 2008.

[22] W. Zhou, H. Li, Y. Lu, and Q. Tian, "Principal visual word discovery for automatic license plate detection," *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 4269–4279, 2012.

[23] M. Piccardi, "Background subtraction techniques: A review," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 2004.

[24] T. Bouwmans, "Recent advanced statistical background modeling for foreground detection: A systematic survey," *Recent Patents on Computer Science*, vol. 4, no. 3, pp. 147–176, 2011.

[25] V. Mahadevan and N. Vasconcelos, "Background subtraction in highly dynamic scenes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[26] X. Ren, T. X. Han, and Z. He, "Ensemble video object cut in highly dynamic scenes," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013.

[27] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, 2000.

[28] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, 2002.

[29] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[30] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Boston, MA, USA: Addison-Wesley Longman, 1992.

[31] R. Kays, S. Tilak, B. Kranstauber, P. A. Jansen, C. Carbone, M. J. Rowcliffe, T. Fountain, J. Eggert, and Z. He, "Monitoring wild animal communities with arrays of motion sensitive camera traps," *International Journal of Research and Reviews in Wireless Sensor Networks*, no. 1, pp. 19–29, 2011.

[32] S. Liao, G. Zhao, V. Kellokumpu, M. Pietikainen, and S. Z. Li, "Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[33] L. Li, W. Huang, I. Y. Gu, and Q. Tian, "Foreground object detection from videos containing complex background," in *Proceedings of ACM International Conference on Multimedia*, 2003.

[34] "Whale FM." [Online]. Available: http://whale.fm/

[35] A. Temko, R. Malkin, C. Zieger, D. Macho, C. Nadeu, and M. Omologo, "Acoustic event detection and classification in smart-room environments: Evaluation of CHIL project systems," *Cough*, vol. 65, no. 48, p. 5, 2006.

[36] A. Temko, "CLEAR 2007 AED evaluation plan and workshop." [Online]. Available: http://isl.ira.uka.de/clear07

[37] X. Zhou, X. Zhuang, M. Liu, H. Tang, M. A. Hasegawa-Johnson, and T. S. Huang, "HMM-based acoustic event detection with AdaBoost feature selection," in *Classification of Events, Activities and Relationships Evaluation and Workshop*, 2007.

[38] K. S. Abouchacra, T. Letowski, and T. Mermagen, "Detection and localization of magazine insertion clicks in various environmental noises," *Military Psychology*, vol. 19, no. 3, pp. 197–216, 2007.

[39] J. A. Smith, J. E. Earis, and A. A. Woodcock, "Establishing a gold standard for manual cough counting: Video versus digital audio recordings," *Cough*, vol. 2, pp. 6:1–6, 2006.

[40] M. A. Hasegawa-Johnson, C. Goudeseune, J. Cole, H. Kaczmarski, H. Kim, S. King, T. Mahrt, J.-T. Huang, X. Zhuang, K.-H. Lin, H. V. Sharma, Z. Li, and T. S. Huang, "Multimodal speech and audio user interfaces for K-12 outreach," in *Proceedings of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 2011.

[41] B. Arons, "Speechskimmer: A system for interactively skimming recorded speech," *ACM Transactions on Computer-Human Interaction*, vol. 4, no. 1, pp. 3–38, 1997.

[42] J. R. Anderson, *Cognitive Psychology and Its Implications*. New York, NY, USA: Worth, 2009.

[43] A. V. Belopolsky, A. F. Kramer, and R. Godijn, "Transfer of information into working memory during attentional capture," *Visual Cognition*, vol. 16, no. 4, pp. 409–418, 2008.

[44] E. B. Goldstein, *Sensation and Perception*. Belmont, CA, USA: Wadsworth, 2010.

[45] C. Goudeseune, "Effective browsing of long audio recordings," in *ACM International Workshop on Interactive Multimedia on Mobile and Portable Devices*, 2012.

[46] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley Interscience, 2006.

[47] K.-H. Lin, X. Zhuang, C. Goudeseune, S. King, M. Hasegawa-Johnson, and T. S. Huang, "Improving faster-than-real-time human acoustic event detection by saliency-maximized audio visualization," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, 2012.

[48] G. Miller, "The magical number seven, plus or minus two: some limits on our capacity for processing information," *Psychological Review*, vol. 63, no. 2, pp. 81–97, 1956.

[49] C. D. Wickens and J. G. Hollands, *Engineering Psychology and Human Performance*. Upper Saddle River, NJ, USA: Prentice Hall, 1999.

[50] H. Jänicke and M. Chen, "A salience-based quality metric for visualization," in *Proceedings of Eurographics / IEEE VGTC conference on Visualization*, 2010.

[51] R. Rosenholtz, A. Dorai, and R. Freeman, "Do predictions of visual perception aid design?" *ACM Transactions on Applied Perception*, vol. 8, no. 2, pp. 1–20, 2011.

[52] D. Walther and C. Koch, "Modeling attention to salient proto-objects," *Neural Networks*, vol. 19, no. 9, pp. 1395–1407, 2006.

[53] S. Frintrop, E. Rome, and H. I. Christensen, "Computational visual attention systems and their cognitive foundations: A survey," *ACM Transactions on Applied Perception*, vol. 7, no. 1, pp. 6:1–6:39, Jan. 2010.

[54] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 185–207, 2013.

[55] A. Bovik, *The Essential Guide to Image Processing*. Waltham, MA, USA: Academic Press, 2009.

[56] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[57] J. S. Lim, *Two-Dimensional Signal and Image Processing*. Upper Saddle River, NJ, USA: Prentice Hall, 1990.

[58] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization." *Nature*, vol. 401, no. 6755, pp. 788–91, 1999.

[59] M. Berry, M. Browne, A. Langville, V. Pauca, and R. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 155–173, 2007.

[60] P. Smaragdis and J. C. Brown, "Nonnegative matrix factorization for polyphonic music transcription," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2003.

[61] P. Smaragdis, "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs," in *International Symposium on Independent Component Analysis and Blind Source Separation*, 2004.

[62] J. Carletta, "Unleashing the killer corpus: Experiences in creating the multi-everything ami meeting corpus," *Language Resources and Evaluation*, vol. 41, no. 2, pp. 181–190, 2007.

[63] C. J. V. Rijsbergen, *Information Retrieval*. Newton, MA, USA: Butterworth-Heinemann, 1979.

[64] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2012.

[65] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.